INSO
16386-4
1st.Edition
Jul.2013

جمهوری اسلامی ایران
Islamic Republic of Iran
سازمان ملی استاندارد ایران
Iranian National Standardization Organization

استاندارد ملی ایران
۱۶۳۸۶ـ۴
چاپ اول
مرداد ۱۳۹۲

کارت‌های شناسایی ـ واسط‌های برنامه نویسی کارت دارای مدار مجتمع ـ قسمت ۴ :

مدیریت واسط برنامه نویسی کاربردی (API)

Identification cards –Integrated circuit card programming interfaces Part 4 : Application programming interface (API) administration

ICS:35.240.15

به نام خدا

آشنایی باسازمان ملی استاندارد ایران

مؤسسهٔاستاندارد و تحقیقات صنعتی ایران به موجب بندیک مادهٔ۳ قانون اصلاح قوانین ومقررات مؤسسهٔ استانداردو تحقیقات صنعتی ایران،مصوب بهمن ماه۱۳۷۱تنها مرجع رسمی کشوراست که وظیفه تعیین،تدوین ونشر استانداردهای ملی (رسمی) ایران رابه عهده دارد.

نام موسسه استاندارد و تحقیقات صنعتی ایران به موجب یکصد و پنجاه و دومین جلسه شورای عالی اداری مورخ ۹۰/۶/۲۹ به سازمان ملی استاندارد ایران تغییر و طی نامه شماره ۲۰۶/۳۵۸۳۸ مورخ ۹۰/۷/۲۴ جهت اجرا ابلاغ شده است .

تدوین استاندارد درحوزه های مختلف درکمیسیون های فنی مرکب ازکارشناسان سازمان ،صاحبنظران مراکز و مؤسسات علمی،پژوهشی،تولیدی واقتصادی آگاه ومرتبط انجام میشود وکوششی همگام با مصالح ملی وباتوجه به شرایط تولیدی، فناوری وتجاری است که از مشارکت آگاهانه ومنصفانهٔ صاحبان حق و نفع ، شامل تولیدکنندگان ، مصرف‌کنندگان، صادرکنندگان و واردکنندگان،مراکز علمی وتخصصی ،نهادها ، سازمان های دولتی و غیر دولتی حاصل میشود.پیش نویس استانداردهای ملی ایران برای نظرخواهی به مراجع ذینفع و اعضای کمیسیون های فنی مربوط ارسال می‌شود و پس از دریافت نظرها و پیشنهادها درکمیتهٔ ملی مرتبط با آن رشته طرح و در صورت تصویب به عنوان استاندارد ملی (رسمی) ایران چاپ و منتشر میشود.

پیش نویس استانداردهایی که مؤسسات و سازمان های علاقه مند وذیصلاح نیز با رعایت ضوابط تعیین شده تهیه میکنند درکمیتهٔ ملی طرح و بررسی و در صورت تصویب ،به عنوان استاندارد ملی ایران چاپ و منتشر میشود .بدین ترتیب، استانداردهایی ملی تلقی میشوندکه براساس مفاد نوشته شده در استاندارد ملی ایران شمارهٔ ۵ تدوین و در کمیتهٔ ملی استاندارد مربوط که سازمان ملی استان دارد ایران تشکیل می‌دهد به تصویب رسیده باشد.

سازمان ملی استانداردایران از اعضای اصلی سازمان بین المللی استاندارد (ISO)[1]،کمیسیون بین المللی الکتروتکنیک (IEC)[2]وسازمان بین المللی اندازه شناسی قانونی(OIML)[3] است و به عنوان تنها رابط[4]کمیسیون کدکس غذایی (CAC)[5]درکشورفعالیت می کند. درتدوین استانداردهای ملی ایران ضمن توجه به شرایط کلی ونیازمندی های خاص کشور ،ازآخرین پیشرفت های علمی،فنی وصنعتی جهان و استانداردهای بین المللی بهره‌گیری می‌شود .

سازمان ملی استانداردایران میتواند با رعایت موازین پیش بینی شده در قانون ، برای حمایت از مصرف کنندگان، حفظ سلامت و ایمنی فردی و عمومی ، حصول اطمینان از کیفیت محصولات و ملاحظات زیست محیطی و اقتصادی ،اجرای بعضی ازاستانداردهای ملی ایران را برای محصولات تولیدی داخل کشور و /یا اقلا م وارداتی ، با تصویب شورای عالی استاندارد، اجباری نماید. سازمان می تواند به منظور حفظ بازارهای بین المللی برای محصولات کشور،اجرای استاندارد کالاهای صادراتی و درجه‌بندی آن را اجباری نماید .همچنین برای اطمینان بخشیدن به استفاده کنندگان از خدمات سازمان ها و مؤسسات فعال در زمینهٔ مشاوره،آموزش،بازرسی،ممیزی و صدورگواهی سیستم های مدیریت کیفیت ومدیریت زیست محیطی،آزمایشگاه ها و مراکز کالیبراسیون (واسنجی) وسایل سنجش،سازمان ملی استاندارد ایران این گونه سازمان ها و مؤسسات را بر اساس ضوابط نظام تأیید صلاحیت ایران ارزیابی میکند و در صورت احراز شرایط لازم،گواهینامهٔ تأیید صلاحیت به آن ها اعطا و بر عملکرد آن ها نظارت می کند.ترویج دستگاه بین المللی یکاها ،کالیبراسیون (واسنجی)وسایل سنجش،تعیین عیار فلزات گران بها و انجام تحقیقات کاربردی برای ارتقای سطح استانداردهای ملی ایران از دیگر وظایف این سازمان است.

1- International Organization for Standardization
2 - International Electrotechnical Commission
3- International Organization of Legal Metrology (OrganisationInternationale de MetrologieLegale)
4 - Contact point
5 - Codex Alimentarius Commission

کمیسیون فنی تدوین استاندارد

"کارت‌های شناسایی ـ واسط‌های برنامه نویسی کارت دارای مدار مجتمع ـ

قسمت ۴ :

مدیریت واسط برنامه نویسی کاربردی (API) "

رئیس:	**سمت و/ یا نمایندگی**
تهرانی طریقت، محمدابراهیم (کارشناسی ارشد مدیریت فناوری اطلاعات)	مشاور ریاست سازمان ثبت احوال و قائم مقام مجری طرح کارت ملی هوشمند
دبیر:	
داوری تبریزی، بیژن (لیسانس مهندسی صنایع)	مدیر عامل شرکت مهندسی و بهبود کیفیت شریف
اعضاء: (اسامی به ترتیب حروف الفبا)	
بداغی، امیرحسین (کارشناسی ارشد مهندسی الکترونیک)	کارشناس سازمان فناوری اطلاعات
جمیل‌پناه، ناصر (کارشناسی ارشد مدیریت)	کارشناس سازمان فناوری اطلاعات
جهان‌شاه، فرناد (کارشناسی مهندسی نرم‌افزار)	کارشناس شرکت مهندسی و بهبود کیفیت شریف
سعیدی، عذرا (کارشناسی ارشد مهندسی مخابرات)	کارشناس سازمان فناوری اطلاعات
صفرنیا، فتانه (کارشناسی فیزیک)	نماینده حوزه طرح کارت ملی هوشمند سازمان ثبت احوال

زنده نام، مهدی (کارشناسی فناوری اطلاعات)	کارشناس حوزه طرح کارت ملی هوشمند سازمان ثبت احوال
نوروزی‌زاده، حمیرا (کارشناسی مهندسی صنایع)	مدیر پروژه تدوین استاندارد شرکت مهندسی و بهبود کیفیت شریف و کارشناس استاندارد

فهرست مندرجات

عنوان **صفحه**

پیش گفتار

استاندارد " کارت‌های شناسایی ـ واسط‌های برنامه نویسی کارت دارای مدار مجتمع ـ قسمت ۴ : مدیریت واسط برنامه نویسی کاربردی (API)" که پیش‌نویس آن در کمیسیون‌های مربوط توسط شرکت مهندسی و بهبود کیفیت شریف تهیه و تدوین شده است و در صدوشصت‌وسومین اجلاس کمیته ملی استاندارد خدمات مورخ ۹۱/۱۲/۲۱ مورد تصویب قرار گرفته است، اینک به استناد بند یک ماده ۳ قانون اصلاح قوانین و مقررات موسسه استاندارد و تحقیقات صنعتی ایران، مصوب بهمن ماه ۱۳۷۱، به عنوان استاندارد ملی ایران منتشر می شود.

برای حفظ همگامی و هماهنگی با تحولات و پیشرفت‌های ملی و جهانی در زمینه صنایع، علوم و خدمات، استانداردهای ملی ایران در مواقع لزوم تجدید نظر خواهد شد و هر پیشنهادی که برای اصلاح و تکمیل این استانداردها ارائه شود، هنگام تجدید نظر در کمیسیون فنی مربوط مورد توجه قرار خواهد گرفت. بنابراین، باید همواره از آخرین تجدید نظر استانداردهای ملی استفاده کرد.

منبع و مأخذی که برای تهیه این استاندارد مورد استفاده قرار گرفته به شرح زیر است:

ISO 24727-4:2008 + Cor 1:2011, Identification cards –Integrated circuit card programming interfaces – Part 4 : Application programming interface (API) administration

مقدمه

استانداردهای ملی ایران شماره ۱۶۳۸۶ مجموعه‌ای از واسط‌های برنامه‌نویسی برای برهم‌کنش(تعامل)[۱] بین کارت‌های دارای مدار مجتمع و برنامه‌های کاربردی خارجی است که خدمات عمومی[۲] برای مصارف چند بخشی[۳] را شامل می‌شود. سازمان و عملکرد کارت های دارای مدار مجتمع با استاندارد ISO/IEC 7816-4 مطابقت دارد .

این استاندارد، قسمتی از مجموعه استانداردهای ملی ایران ۱۶۸۳۶ می‌باشد .

1 -Interaction
2-Generic services
3 - Multi-sector use

کارت‌های شناسایی ـ واسط‌های برنامه‌نویسی کارت دارای مدار مجتمع ـ

قسمت ۴ :

مدیریت واسط برنامه نویسی کاربردی (API)[1]

۱ هدف و دامنه کاربرد

هدف از تدوین این استاندارد، تعیین مجموعه‌ای از واسط‌های برنامه‌نویسی برای تعامل بین کارت‌های دارای مدار مجتمع و برنامه‌های کاربردی خارجی می‌باشد تا خدمات عمومی برای استفاده چند قسمتی، را دربرگیرد.

این استاندارد، سازوکارهای اتصال و امنیتی بین برنامه کاربردی سرویس‌گیرنده (کارخواه)[2] و برنامه کاربردی کارت را تعیین می‌کند.

این استاندارد، مدیریت API مربوط به پودمان‌های مستقل از خدمت و مستقل از پیاده‌سازی که با استاندارد ملی ایران شماره ۱۶۳۸۶ سازگار هستند را مشخص می‌کند شامل امنیت، که درخواست‌های عمل به یک برنامه کاربردی کارت بخصوص مربوط به یک ICC[3] را امکان‌پذیر می‌سازد به طوری که پس از جفت شدن با مدل داده‌ای و عملیات کشف محتوا، برنامه کاربردی کارت می‌تواند به وسیله برنامه‌های کاربردی سرویس‌گیرنده متعددی مورد استفاده قرار گیرد.

۲ مراجع الزامی

مدارک الزامی زیر حاوی مقرراتی است که در متن این استاندارد ملی ایران به آن‌ها ارجاع داده شده است. بدین ترتیب آن مقررات جزئی از این استاندارد ملی ایران محسوب می شود.

در صورتی که به مدرکی با ذکر تاریخ انتشار ارجاع داده شده باشد،اصلاحیه‌ها و تجدیدنظرهای بعدی آن مورد نظر این استاندارد ملی ایران نیست. در مورد مدارکی که بدون ذکر تاریخ انتشار به آن‌ها ارجاع داده شده است، همواره آخرین تجدید نظر و اصلاحیه های بعدی آن ها مورد نظر است.

استفاده از مراجع زیر برای این استاندارد الزامی است:

۲ ـ ۱ استاندارد ملی ایران شماره ۱ ـ ۱۶۳۸۶، کارت‌های شناسایی ـ واسط‌های برنامه نویسی کارت دارای مدار مجتمع ـ قسمت ۱ :معماری

۲ ـ ۲ استاندارد ملی ایران شماره ۲ ـ ۱۶۳۸۶،، کارت‌های شناسایی ـ واسط‌های برنامه نویسی کارت دارای مدار مجتمع ـ قسمت ۲ : واسط عمومی کارت

۲ ـ ۳ استاندارد ملی ایران شماره ۳ ـ ۱۶۳۸۶،کارت‌های شناسایی ـ واسط‌های برنامه‌نویسی کارت دارای مدار مجتمع ـ قسمت ۳: واسط برنامه کاربردی

۲ ـ ۴ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ ـ۹۷۹۷، فناوری اطلاعات -فنون امنیتی -کدهای احراز هویت پیام (MAC)قسمت ۱ - سازو کارهای استفاده از رمزگذاری بلوکی

2-5 ISO/IEC 7816-4:2005, Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange

---

1-Application Programming Interface(API)
2- Client application
3-Integrated Circuit Cards (ICC)

۳ اصطلاحات و تعاریف

در این استاندارد، علاوه بر اصطلاحات و تعاریف تعیین شده در استاندارد ملی ایران شماره۱ـ ۱۶۳۸۶، اصطلاحات و تعاریف زیر نیز به‌کار می‌رود:

۳-۱

کانال[1]

مسیر فیزیکی که به بیت‌های اطلاعاتی، امکان جابجایی بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت را می‌دهد

۳-۲

مولفه[2]

کد قابل اجرای تشکیل شده از یک لایه پردازشی، که با واسطهای برنامه‌نویسی کاربردی تعریف شده در استاندارد ملی ایران شماره ۱۶۳۸۶، مورد دسترسی قرار می‌گیرد

۳-۳

محرمانه بودن[3]

دسترسی محدود شده به یک سطح تعریف شده از احراز هویت مربوط به هویت متمایزکننده

۳-۴

کانال مظنون[4]

کانالی که ممکن است به پیام‌های اطلاعاتی امکان دهد تا تغییر یابند، از بین بروند، بازپخش شوند یا به وسیله استراق-سمع‌کنندگان، دریافت شوند

۳-۵

نمونه‌سازی[5]

پیاده‌سازی مولفه عملیاتی یا پیاده‌سازی کانال ارتباطی

۳-۶

یکپارچگی[6]

وضعیت ثبات اطلاعات

۳-۷

پشته پروتکل[7] **استاندارد ملی ایران شماره ۱۶۳۸۶**

دنباله‌ای از مولفه‌های پردازشی متصل به وسیله کانال‌های ارتباطی که یک برنامه کاربردی سرویس‌گیرنده را به یک برنامه کاربردی کارت، متصل می‌کنند

---

1 - Channel
2 - Component
3 - Confidentiality
4 - Dubious-channel
5 - Instantiation
6 - Integrity
7 - Protocol stack

۸-۳

کانال اختصاصی[1]

کانالی که به طور ذاتی، جامعیت نقاط پایانی کانال را به همراه محرمانه بودن، جامعیت و اصالت اطلاعات، حفظ می‌کند

۹-۳

پلاتفرم اختصاصی[2]

پلاتفرم محاسباتی که در حین حفظ محرمانه بودن، جامعیت و اصالت اطلاعات، برای ارتباط و تبدیل داده‌ای، مورد اعتماد است

۱۰-۳

پشته اختصاصی[3]

پشته استاندارد ملی ایران شماره ۱۶۳۸۶، که در آن پشته کاملی از برنامه کاربردی سرویس‌گیرنده گرفته تا کارت مدار مجتمعی که حاوی برنامه کاربردی کارت است، روی یک پلاتفرم اختصاصی منفرد، پیاده‌سازی می‌شود

۱۱-۳

خط‌مشی حفاظت مسیر[4]

مشخصه‌ای از خصوصیات امنیتی تمام پلاتفرم‌ها و کانال‌های استفاده‌کننده از استاندارد ملی ایران شماره ۱۶۳۸۶، که برنامه کاربردی سرویس‌گیرنده را به برنامه کاربردی کارت، متصل می‌کنند

۱۲-۳

پروکسی[5]

پیاده‌سازی واسط برنامه‌نویسی کاربردی که درخواست‌های عمل و پارامترها را به یک پیاده‌سازی لایه قرار گرفته در جایی دیگر، منتقل می‌کند

۱۳-۳

TC_API

واسط برنامه‌نویسی کاربردی مورد استفاده به وسیله مولفه‌های پشته استاندارد ملی ایران شماره ۱۶۳۸۶ برای وقوع نمونه‌سازی پشته در یک محیط شبکه

۱۴-۳

کانال مورد اعتماد[6]

کانالی که در خلال فرآیند انتقال، مستقل از خصوصیات سازوکار انتقال یا رسانه، به طور صریح، از محرمانه بودن، جامعیت و اصالت اطلاعات، اطمینان حاصل می‌کند

1- Loyal-channel
2 - Loyal-platform
3 - Loyal-stack
4 - Path-protection-policy
5 - Proxy
6 - Trusted-channel

۳-۱۵

مسیر مورد اعتماد[1]

اتصال بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت که در آن تمام پلاتفرم‌ها و کانال‌ها، خصوصیات امنیتی تعریف شده به وسیله برنامه کاربردی سرویس‌گیرنده را دارا می‌باشد

۴ کوته‌نوشت‌ها

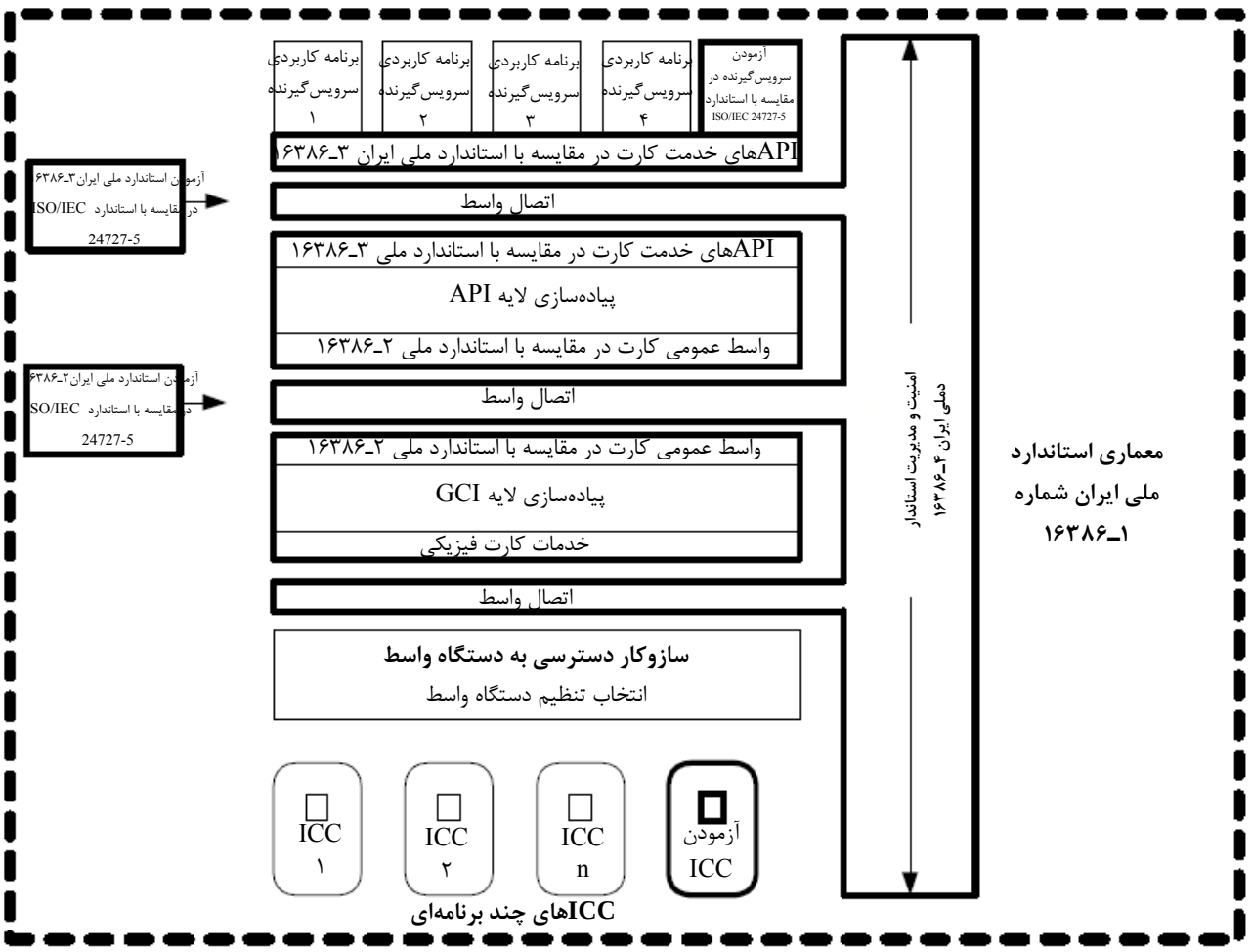
امنیت لایه انتقال	TLS Transport Layer Security
کنترل صحت داده‌های رمزنگاشتی	CC Cryptographic Checksum
متن رمزنگاشتی شده	CG Cryptogram
رمزگشایی	D Decipher
استاندارد رمزنگاری‌داده‌ها	DES Data Encryption Standard
شیء داده‌ای	DO Data Object
رمزنگاشتی	E Encipher
کلید	K Key
طول مورد انتظار	Le expected length
کد احراز هویت پیام	MAC Message Authentication Code
شمارنده ترتیب ارسال	SSC Send Sequence Counter
واسط برنامه‌نویسی کاربردی	API Application Programming Interface
واحد داده‌ای پروتکل کاربردی	APDU Application Protocol Data Unit

۵ تخصصی‌سازی معماری

استانداردهای ملی ایران شماره۱ـ۱۶۳۸۶، ۲ـ۱۶۳۸۶و۳ـ۱۶۳۸۶، یک معماری، واسط برنامه‌نویسی کاربردی، و یک پروتکل و ساختار پیام ارتباطی مجموعه فرمان[2] تعریف می‌نماید که به وسیله آن، یک برنامه کاربردی سرویس‌گیرنده می‌تواند به اطلاعات و خدمات محاسباتی یک برنامه کاربردی کارت، دسترسی یابد. هدف استاندارد ملی ایران شماره ۱۶۳۸۶ دست یافتن به تعامل‌پذیری در میان پیاده‌سازی‌های گوناگون برنامه‌های کاربردی کارت و برنامه‌های کاربردی سرویس‌گیرنده است. استاندارد ملی ایران شماره۱ـ۱۶۳۸۶، معماری جامع استاندارد ملی ایران شماره ۱۶۳۸۶ را مشخص می‌کند. استاندارد ملی ایران شماره۲ـ۱۶۳۸۶، یک مجموعه درخواست عمومی را مشخص می‌کند که به وسیله آن، خدمات برنامه کاربردی کارت ممکن است مورد دسترسی قرار گیرند. استاندارد ملی ایران شماره ۳ـ۱۶۳۸۶، واسط برنامه کاربردی را مشخص می‌نماید که به وسیله آن، برنامه‌های کاربردی سرویس‌گیرنده باید به خدمات فراهم شده به وسیله برنامه‌های کاربردی کارت، دسترسی یابد. بررسی کلی معماری استاندارد ملی ایران شماره ۱۶۳۸۶ همان‌گونه که در شکل ۱ به تصویر کشیده شده است، پیاده‌سازی‌های مختلفی را نشان می‌دهد که می‌توانند برای تطابق موفقیت‌آمیز با رویه‌های آزمودن که در استاندارد ملی ایران شماره ۵ـ۱۶۳۸۶مشخص خواهند شد، این استاندارد را با جزئیات کافی، برآورده نمایند.

---

1 - Trusted-path
2 - Command-set



شکل ۱- تخصصی‌سازی معماری استاندارد ملی ایران شماره ۱۶۳۸۶

این استاندارد، آن نمونه‌سازی پشته و رویه‌های عملیاتی را به طور مفصل بیان می‌کند که آماده‌سازی و استفاده از یک برنامه کاربردی کارت برای فراهم کردن انباره اطلاعات، بازیابی و پردازش مربوطه برای برنامه‌های کاربردی سرویس‌گیرنده، را فراهم می‌کند.

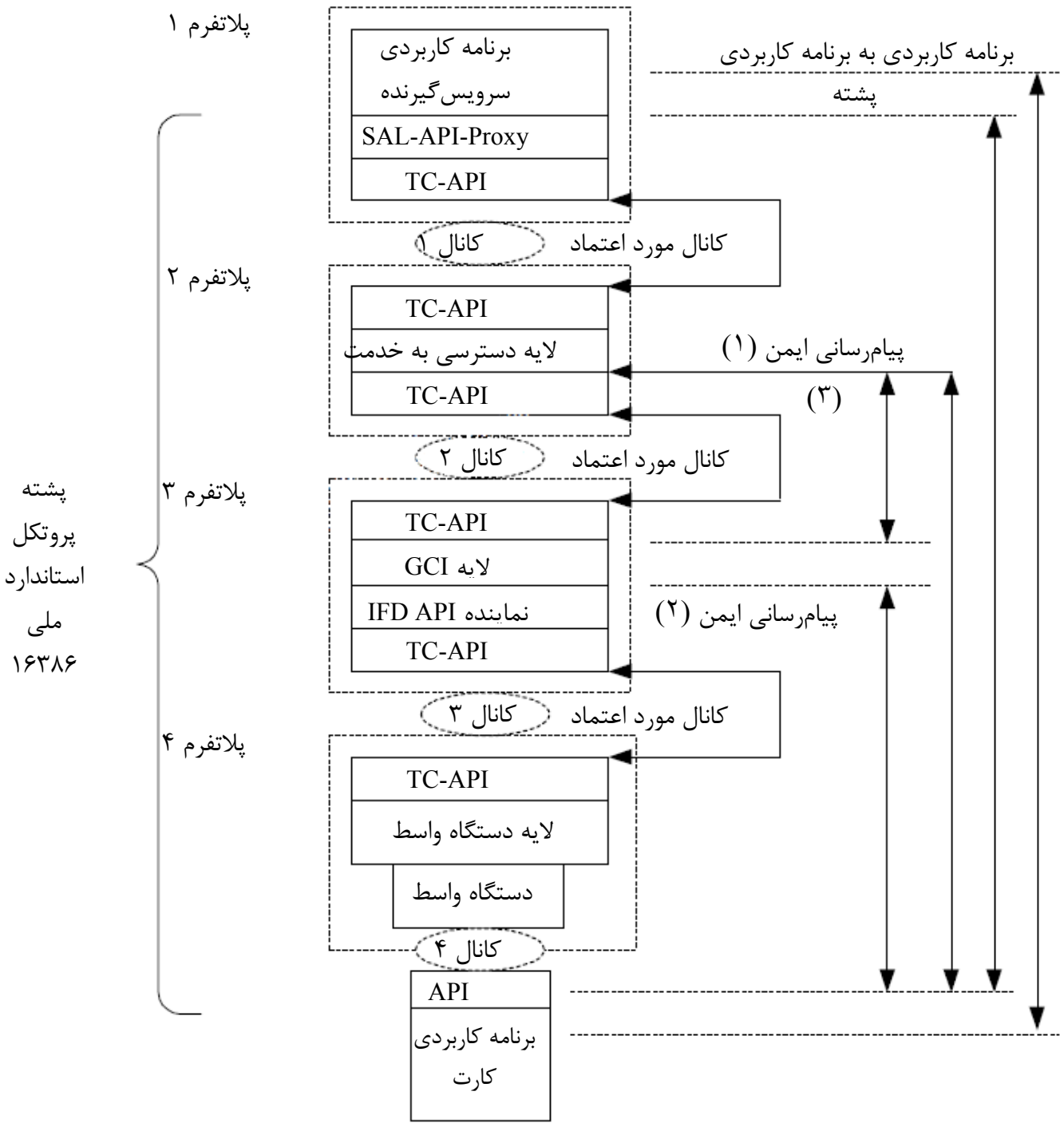
این استاندارد، یک روش پیاده‌سازی مشخص را الزام نمی‌نماید، ولی تعریف مفصلی از سازماندهی اطلاعات و محتوایی که قرار است به وسیله هر پیاده‌سازی سازگار، پشتیبانی شود، را فراهم می‌کند. یک پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ باید حداقل یکی از تنظیمات تعریف شده در بند ۵ را نمونه‌سازی کند.

این بند، تنظیمات نمونه‌سازی گوناگونی را از پشته پروتکل استاندارد ملی ایران شماره ۱۶۳۸۶، مشخص می‌نماید. طیف تنظیمات، از پشته شبکه کامل تا پشته اختصاصی متغیر است. پشته ICC غیر شفاف، تنظیمی است که در آن نمونه‌سازی استاندارد ملی ایران شماره۲ـ۱۶۳۸۶باید به شدت به برنامه‌های کاربردی کارتی که می‌تواند پشتیبانی نماید، وابسته باشد؛ منظور از به شدت وابسته این است که اتصال به برنامه کاربردی کارت حاوی ICC به وسیله کد مخصوص سیستم عامل برای دسترسی به یک دستگاه واسط، درون نمونه‌سازی استاندارد ملی ایران شماره۲ـ۱۶۳۸۶قرار دارد. پشته اختصاصی راه دور، تنظیمی را مدنظر قرار می‌دهد که در آن یک پشته اختصاصی باید روی پلاتفرمی به کار گرفته شود که دور از برنامه کاربردی سرویس‌گیرنده، قرار دارد. پشته مقیم ICC [1]، برنامه کاربردی کارتی را مدنظر قرار می‌دهد که باید پیاده‌سازی لایه استاندارد ملی ایران شماره۳ـ۱۶۳۸۶ را پشتیبانی کند. در پایان، پشته ICC راه دور، در مورد پشته‌ای بحث می‌نماید که در آن، اتصال فیزیکی ICC باید به پلاتفرمی متفاوت از بقیه پشته، ایجاد شود.

1 - ICC-Resident

این نکته مورد توجه است که تمام قسمت‌های استاندارد ملی ایران شماره ۱۶۳۸۶ نسبت به سازوکار اتصال داخلی فیزیکی مورد استفاده برای تکمیل کانال(های) ارتباطی از برنامه کاربردی سرویس‌گیرنده به برنامه کاربردی کارت، خنثی است. به طور متعاقب، مرجع‌ها به ICC بهتر است به صورت برابر از لحاظ قابلیت استفاده برای PICC یا برای برنامه‌های کاربردی کارت مقیم غیرکارتی تفسیر شوند و این که مرجع‌ها به IFD، از لحاظ قابلیت استفاده برای PCD یا برنامه کاربردی دیگر شامل واسط‌های پلاتفرم، برابر است.

شکل ۲، اجزاء عمومی یک پشته پروتکل استاندارد ملی ایران شماره ۱۶۳۸۶ را نشان می‌دهد.



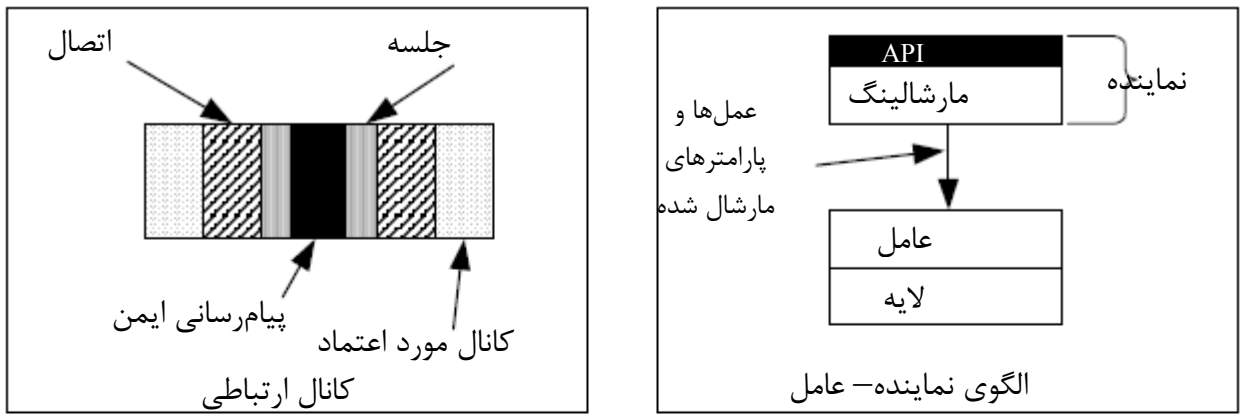
شکل ۲- اجزاء عمومی پشته استاندارد ملی ایران شماره ۱۶۳۸۶

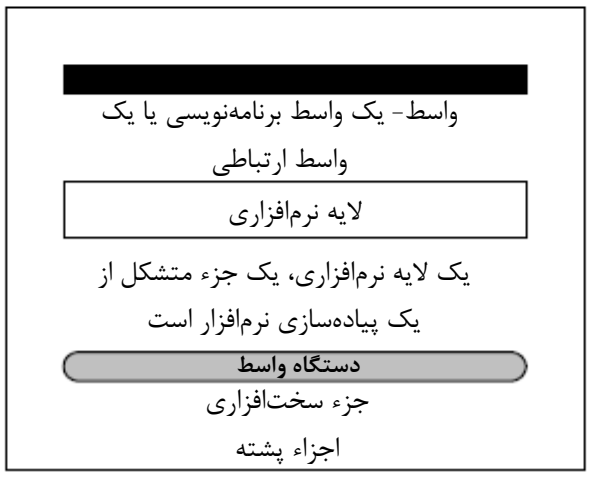
این اجزاء، یک تنظیم پشته عمومی، تعریف می‌کند. این تنظیم پشته عمومی، به یک پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ امکان می‌دهد که در میان محدوده‌ای از یک تا چهار پلاتفرم رایانه‌ای مجزا، بخش‌بندی شود. در شکل ۲، این پلاتفرم‌ها به ترتیب به صورت پلاتفرم ۱، ۲، ۳ و ۴ نامگذاری شده‌اند.

در یک پشته کاملا بخش‌بندی شده، برای متصل کردن مولفه‌هایی که پشته را تشکیل داده‌اند، چهار کانال ارتباطی مجزا مورد نیاز است. این کانال‌ها به ترتیب به صورت کانال ۱، ۲، ۳ و ۴ نامگذاری شده‌اند. در بندهای ۵-۱ تا ۵-۶، شش تنظیم پشته مجزا، مشخص می‌شوند. این‌ها، مجموعه پشته‌های مجاز سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ را تشکیل می‌دهد. یک نمونه‌سازی پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ باید شامل حداقل یکی از این شش مشخصات پشته باشد. یک نمونه‌سازی پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ ممکن است شامل بیش از یکی از این شش تنظیمات پشته باشد.

شکل ۳، یک راهنمای توصیفی برای استفاده در تفسیر جزئیات بقیه شکل‌های بند ۵، ارائه می‌کند.

راهنمای شکل





شکل ۳- راهنمای شکل‌های بعدی

یک جفت (فرآیندهای) نماینده و عامل، به وسیله تبدیل درخواست‌های عمل و پارامترهای مربوطه توسط یک توصیف استاندارد، امکان توسعه یک API از یک نقطه در پشته به نقطه‌ای دیگر را می‌دهد؛ رویه‌ای که به عنوان «مارشالینگ[1]» شناخته می‌شود.

1 - Marshalling

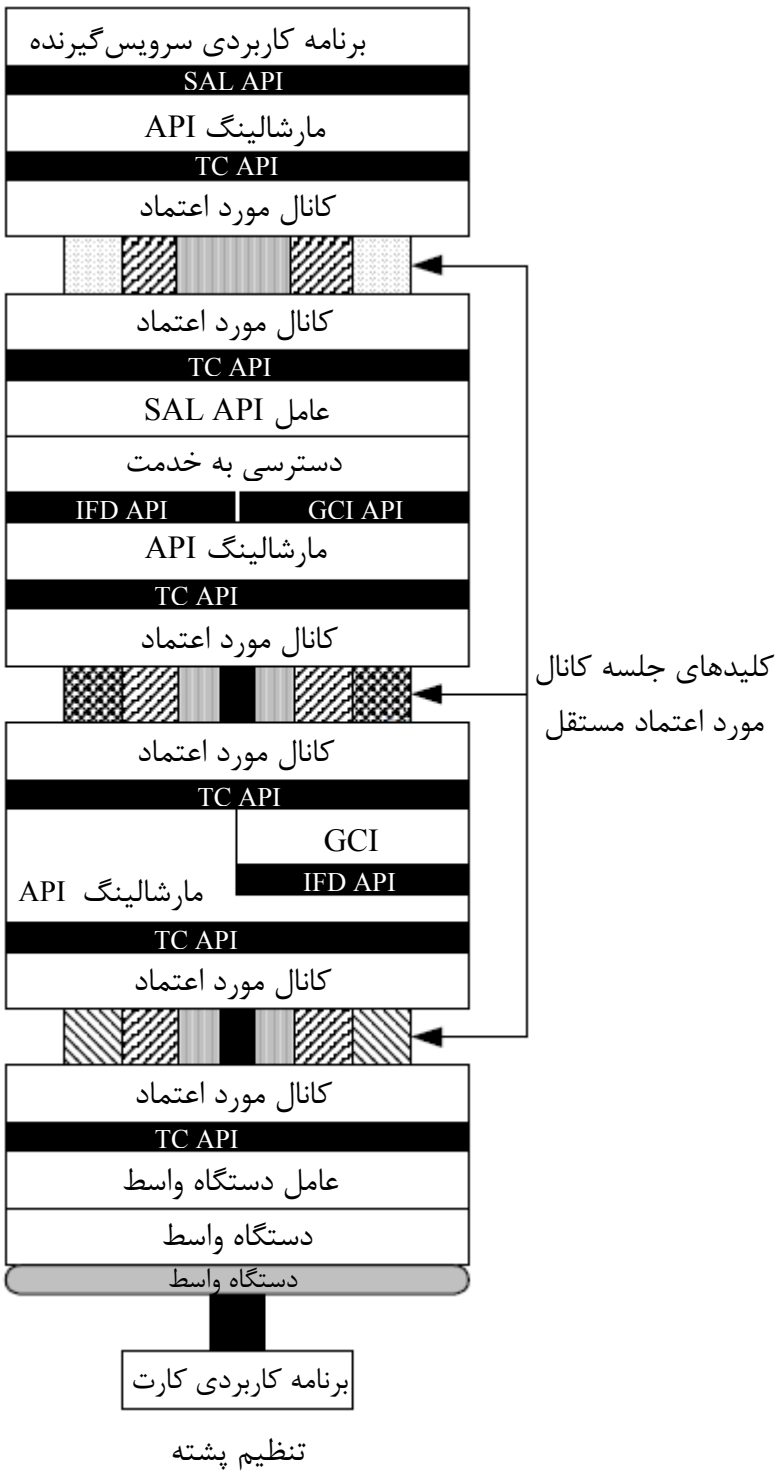
۵-۱ پشته شبکه کامل

یک ارتباط داخلی عمومی بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت، به هر مولفه پشته امکان می‌دهد که همان‌گونه که در شکل ۴ نشان داده شده است، از طریق یک اتصال شبکه، به مولفه‌های مجاور خود، متصل شود.

تنظیم پشته شبکه کامل استاندارد ملی ایران شماره ۱۶۳۸۶، بخش‌بندی پشته استاندارد ملی ایران شماره ۱۶۳۸۶ را به مولفه‌های تشکیل‌دهنده تعامل‌پذیر آن، منحصر می‌کند. در حالی که نامحتمل است که چنین تنظیم پشته‌ای در یک محیط عملیاتی عادی، استفاده شود، به وسیله آزمودن سازگاری این تنظیم، یک سطح بسیار دقیق از تعامل‌پذیری مولفه، می‌تواند تایید شود.

تحقق یک پشته شبکه کامل باید به وسیله تنظیمات شبکه ایستای مولفه‌های گوناگون، به وقوع بپیوندد. در استاندارد ملی ایران شماره ۱۶۳۸۶، فراخوانی فرآیند پویای مولفه‌ها، ضروری نیست. برقراری مشخصات امنیتی انتها تا انتها، باید به وسیله پارامترهای منتقل شده از طریق API استانداردملی ایران شماره۳ـ۱۶۳۸۶ واقع شود. نمونه‌سازی و عمل پشته ممکن است متعاقبا با توجه به واسط استاندارد ملی ایران شماره۲ـ ۱۶۳۸۶، انتقال پارامتر از خارج از محدوده، را دربرگیرد، بخصوص در برقراری کلیدهای مناسب برای فعال کردن پیام‌رسانی ایمن بعدی بین پیاده‌سازی لایه استاندارد ملی ایران ۲ـ۱۶۳۸۶و برنامه کاربردی کارت.

برقراری پشته متناظر و مشخصات امنیتی اتصال (وجلسه) باید به وسیله نمونه‌سازی‌های تمام سایر تنظیمات پشته مشخص شده در زیربندهای متعدد بند ۵، برآورده شود.



شکل ۴- اتصالات شبکه‌ای بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت

در شکل ۴، اتصال برنامه کاربردی سرویس‌گیرنده به نماینده SAL-API به وسیله یک کانال اختصاصی، واقع می‌شود. تمام کانال‌های ارتباطی مشخص شده دیگر در این شکل، کانال‌های مظنون هستند. اگر سیاست حفاظت مسیر فراخوانی شده به وسیله برنامه کاربردی سرویس‌گیرنده، یک سطح امنیتی فزاینده فراتر از یک کانال مظنون را برای هر کانال ارتباطی، مشخص نماید آنگاه این سطح باید با استفاده از کانال‌های مورد اعتماد یا در صورت امکان، با استفاده از پیام‌رسانی ایمن، به دست آید.

اگر قرار باشد که تمام پشته به صورت یک دسته ایمن فراتر از یک سطح امنیت ذاتی در نظر گرفته شود، پیاده‌سازی استاندارد ملی ایران شماره ۳ـ۱۶۳۸۶، پیاده‌سازی استاندارد ملی ایران شماره ۲ـ۱۶۳۸۶و پیاده‌سازی‌های واسطه‌های کارت، تماما مولفه‌هایی را تشکیل می‌دهند که باید روی پلاتفرم‌های مورد اعتماد، واقع شوند. در این مورد، در بند ۶ بحث می‌شود.

۵-۲ پشته اختصاصی

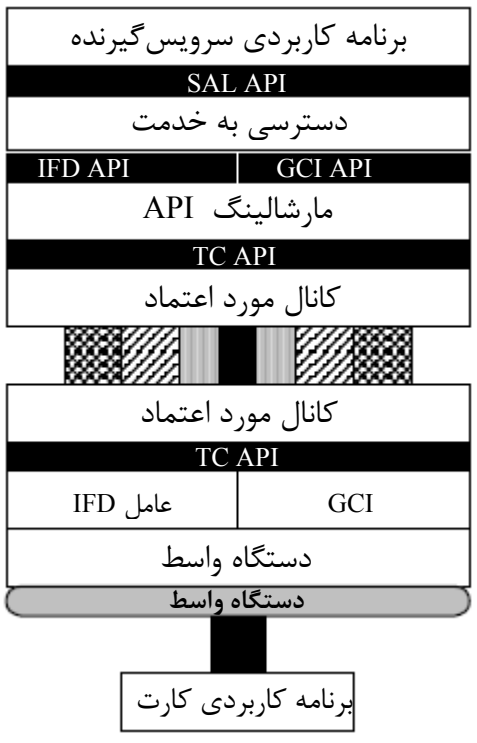
همان‌گونه که در شکل ۵ نشان داده شده، یک پشته اختصاصی، یک پیاده‌سازی پشته کامل استاندارد ملی ایران شماره ۱۶۳۸۶ روی یک پلاتفرم اختصاصی است، در نتیجه، استفاده از کانال‌های اختصاصی برای تمام اتصال‌ها به غیر از هر اتصال به یک ICC از طریق یک دستگاه واسط. برای ارتباط از طریق دستگاه واسط با برنامه کاربردی کارت با استفاده از پیام‌رسانی ایمن، برای لایه API دستگاه واسط مختص سیستم عامل، باید یک دسته امنیتی توسعه یافته، حاصل شود.

شکل ۵- پیاده‌سازی‌های اختصاصی لایه‌های استانداردهای ملی ایران شماره۲ـ ۱۶۳۸۶و ۳ـ۱۶۳۸۶

یک پشته اختصاصی باید همان‌گونه که در بند ۶ تعریف شده، باعث وقوع یک دسته سیاست حفاظت مسیر ذاتی شود، یا با تنها کانال مظنون ممکن، که اتصال این پشته به برنامه کاربردی کارت از طریق یک دستگاه واسط با استفاده از پیام‌رسانی ایمن، می‌باشد، به صورت اشاره شده در بالا، در یک محیط اختصاصی، به طور کامل نمونه‌سازی شود. هنگام اجرای یک کانال ایمن درون یک پشته اختصاصی، استفاده از TC_API اجباری نیست.

۵-۳ پشته ICC غیر شفاف

یک پشته ICC غیر شفاف، نمونه‌سازی لایه استاندارد ملی ایران شماره۲ـ۱۶۳۸۶و نمونه‌سازی لایه دستگاه واسط را در یک مولفه واحد، مجتمع می‌کند. این مولفه، اتصال خاص سیستم عامل را از طریق یک دستگاه واسط با برنامه کاربردی کارت، در بر می‌گیرد. این مولفه باید به عنوان یک فرآیند قابل دسترسی از طریق شبکه ایستا وجود داشته باشد. فرآیندی که در مذاکره یک کانال مورد اعتماد، لایه کانال مورد اعتماد آن، باید خودش را به عنوان مولفه سرویس‌دهنده، معرفی نماید (به IETF RFC 2246 مراجعه شود).

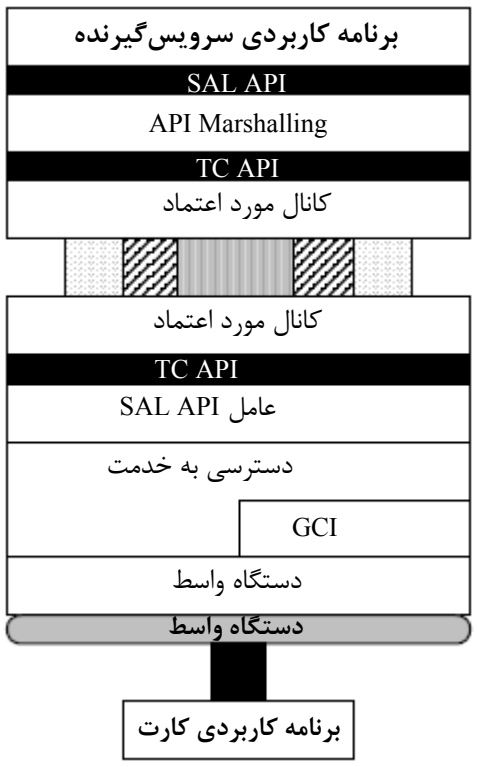


شکل ۶- پشته ICC غیر شفاف

کلیدهایی که به وسیله آن‌ها پیام‌رسانی ایمن بین نمونه‌سازی لایه استاندارد ملی ایران شماره ۲ـ۱۶۳۸۶و برنامه کاربردی کارت واقع می‌شود باید همان‌گونه که در استاندارد ملی ایران شماره۲ـ۱۶۳۸۶مشخص شده، از طریق جزء رویه‌ای فراخوانی شده در خلال عمل راه‌اندازی لایه استاندارد ملی ایران شماره ۲ـ۱۶۳۸۶، به دست آید.

۴-۵ پشته اختصاصی راه دور

یک پشته اختصاصی راه دور، یک پشته اختصاصی را به دو مولفه مجزا بخش‌بندی می‌کند که به برنامه کاربردی سرویس‌گیرنده امکان می‌دهد روی یک شبکه، از پشته استاندارد ملی ایران شماره ۱۶۳۸۶ پشتیبانی‌کننده آن و برنامه کاربردی کارتی که با آن ارتباط دارد، جابجا شود. یک مولفه نماینده SAL-API به طور مستقیم به برنامه کاربردی سرویس‌گیرنده، متصل است و همان‌گونه که در شکل ۷ نشان داده شده، این نماینده به وسیله یک کانال مورد اعتماد با بقیه پشته اختصاصی ارتباط دارد.



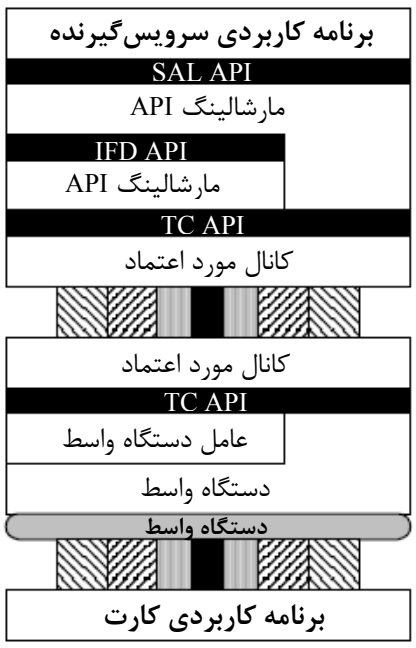
تنظیم پشته

شکل ۷- پشته اختصاصی راه دور

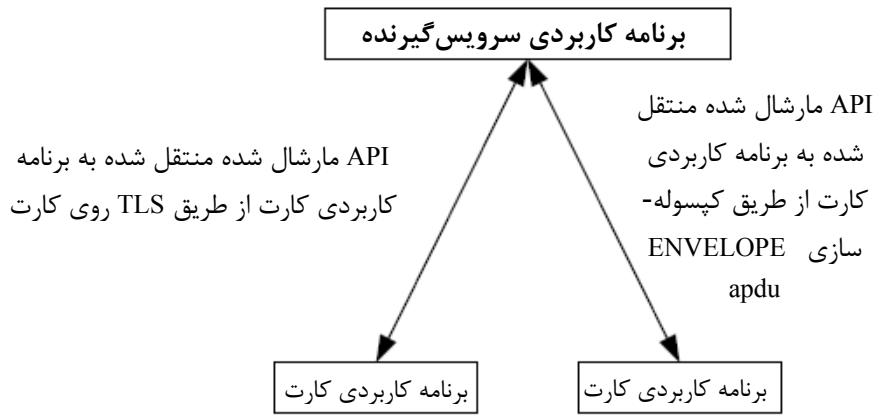
لایه استاندارد ملی ایران شماره۳ـ۱۶۳۸۶ و مولفه لایه استاندارد ملی ایران شماره ۲ـ۱۶۳۸۶باید در زمان نمونه‌سازی پشته اختصاصی راه دور به وسیله عمل برنامه کاربردی سرویس‌گیرنده، به عنوان یک مولفه قابل دسترسی شبکه ایستا، موجود باشد. این مولفه باید به عنوان یک فرآیند قابل دسترسی شبکه ایستا وجود داشته باشد. فرآیندی که در مذاکره یک کانال مورد اعتماد با مولفه برنامه کاربردی سرویس‌گیرنده، لایه کانال مورد اعتماد آن، باید خودش را به عنوان مولفه سرویس‌دهنده، معرفی نماید (به IETF RFC 2246 مراجعه شود).

۵-۵ پشته مقیم ICC

یک پشته مقیم ICC، نمونه‌سازی پشته کامل استاندارد ملی ایران شماره ۱۶۳۸۶ را درون یک برنامه کاربردی کارت فراهم می‌کند. تنها مولفه‌های خارج از کارت، نماینده SAL-API و لایه دستگاه واسط هستند که همان‌گونه که در شکل ۸ نشان داده شده، ارتباط ترکیبی، معنایی و اتصال فیزیکی بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت را فراهم می‌کنند. همان‌گونه که در شکل ۸ نشان داده شده، هر یک از دو مسیر مجزا، می‌تواند برای انتقال API که مارشال شده است، به برنامه کاربردی سرویس‌گیرنده، مورد استفاده قرار گیرد. مسیر مناسب، از طریق آدرس‌دهی که به وسیله درخواست CardApplicationPath تعریف شده در استاندارد ملی ایران شماره۳ـ۱۶۳۸۶، به برنامه کاربردی سرویس-گیرنده، برگردانده می‌شود، تعیین می‌گردد.



تنظیم پشته



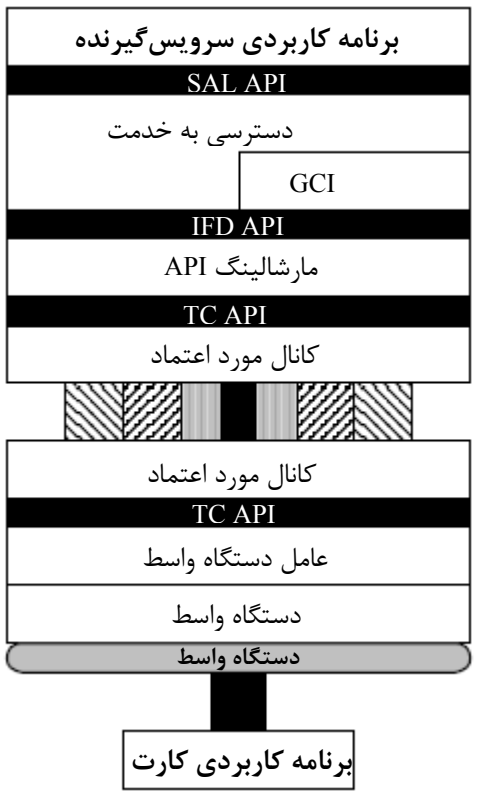
سازوکارهای دو مسیره بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت

شکل ۸- پشته مقیم ICC

نمونه‌سازی لایه کانال مورد اعتماد متعلق به برنامه کاربردی کارت در زمان نمونه‌سازی پشته استاندارد ملی ایران شماره ۱۶۳۸۶ به‌وسیله عمل برنامه کاربردی سرویس‌گیرنده، باید به عنوان یک مولفه قابل دسترسی از طریق شبکه ایستا، موجود باشد. این مولفه باید به عنوان یک فرآیند قابل دسترسی از طریق شبکه ایستا وجود داشته باشد. فرآیندی که در مذاکره یک کانال مورد اعتماد با مولفه برنامه کاربردی سرویس‌گیرنده، لایه کانال مورد اعتماد آن، باید خودش را به عنوان هستار سرویس‌دهنده، معرفی نماید (به IETF RFC 2246 مراجعه شود). پشتیبانی از TLS به وسیله ICC، در استاندارد ISO/IEC 7816 تعریف نشده است. اجزاء مارشال شده SAL API نیز ممکن است به‌صورتی‌که در استاندارد ISO/IEC 7816-4 تعریف شده و در شکل ۸ نشان داده شده است، به صورت کپسوله شده درون یک ENVELOPE apdu به برنامه کاربردی کارت، منتقل شود.

۵-۶ پشته ICC راه دور

شکل ۹، یک پشته ICC راه دور را نشان می‌دهد. در این تنظیم، پشته کامل استاندارد ملی ایران شماره ۱۶۳۸۶، روی همان پلاتفرم برنامه کاربردی سرویس‌گیرنده، پیاده‌سازی می‌شود. نقطه اتصال فیزیکی برنامه کاربردی کارت، ممکن است روی یک پلاتفرم متفاوت از پلاتفرم برنامه کاربردی سرویس‌گیرنده، با یک کانال مورد اعتماد که اتصال بین دو مولفه پشته کامل را فراهم می‌کند، ارائه شود.



تنظیم پشته

شکل ۹- پشته ICC راه دور

در این تنظیم، نمونه‌سازی لایه API دستگاه واسط باید در زمان نمونه‌سازی پشته به وسیله عمل برنامه کاربردی سرویس-گیرنده، به عنوان یک مولفه قابل دسترسی از طریق شبکه ایستا، موجود باشد. این مولفه باید به عنوان یک فرآیند قابل دسترسی از طریق شبکه ایستا وجود داشته باشد. فرآیندی که در مذاکره یک کانال مورد اعتماد با مولفه برنامه کاربردی سرویس‌گیرنده، لایه کانال مورد اعتماد آن، باید خودش را به عنوان هستار سرویس‌دهنده، معرفی نماید (به IETF RFC 2246 مراجعه شود).

۶ معماری امنیت

نمونه‌سازی یک برنامه کاربردی کارت، شامل قواعد کنترل دسترسی مورد استفاده در برنامه کاربردی کارت برای ایمن نمودن ذخیره‌سازی اطلاعات و دسترسی به فرآیند محاسباتی، باید به وسیله یک برنامه کاربردی سرویس‌گیرنده از طریق API استاندارد ملی ایران شماره۳ـ۱۶۳۸۶ یا به‌وسیله یک رویه که از لحاظ عملیاتی، کاملا معادل باشد، اجرا شود. اتصال به یک پشته استاندارد ملی ایران شماره ۱۶۳۸۶ باید به وسیله یک برنامه کاربردی سرویس‌گیرنده از طریق API استاندارد ملی ایران شماره۳ـ۱۶۳۸۶ واقع شود. تنظیمات پشته‌ای که قرار است به وسیله یک برنامه کاربردی سرویس‌گیرنده مورد دسترسی قرار گیرد، باید قبل از دسترسی به وسیله یک برنامه کاربردی سرویس‌گیرنده، نمونه‌سازی شود. باید مشخصات امنیتی هر پشته در دسترس، به‌وسیله پاسخ درخواست API CardApplicationPath استاندارد ملی ایران شماره۳ـ۱۶۳۸۶، از لایه استاندارد ملی ایران شماره۳ـ۱۶۳۸۶به برنامه کاربردی سرویس‌گیرنده، منتقل شود. این باید حق انحصاری برنامه کاربردی سرویس‌گیرنده باشد که تنظیم پشته قابل قبولی را انتخاب نماید که دسترسی به برنامه کاربردی کارت مطلوب را

فراهم می‌کند. بند ۶-۱ معانی که قرار است برای توصیف مشخصات امنیتی پشته، مورد استفاده قرار گیرند را تعریف می‌کند.

۶-۱ سیاست حفاظت مسیر

یک پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ باید از طریق مولفه‌های پشته میانی، باعث وقوع یک کانال ارتباطی بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت شود. همان¬گونه که در شکل ۲ نشان داده شده، این مولفه‌ها و کانال‌های مجاور، چندین بخش مجزا را تشکیل می‌دهند. هر یک از این بخش‌ها باید بر اساس گستره‌ای از سازوکارهای ارتباطی باشد که هر یک دارای مجموعه پروتکل‌های مخصوص به خود است که به وسیله تنظیم پشته، مشخص می‌شود. بنابراین، مشخصات امنیتی مربوط به اطلاعات در جریان بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت، باید به سازوکارهای (امنیتی) مشخص شده به وسیله سیاست حفاظت مسیر برقرار شده برای هر تنظیم پشته، مقید شود. بدون توجه به تنظیم پشته، تنها نقاط دسترسی به کانال، باید برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت باشد. مشخصه سیاست حفاظت مسیر باید به وسیله API استانداردملی ایران شماره ۱۶۳۸۶ به عنوان یک پارامتر درخواست CardApplicationPath متعلق به خدمت اتصال، به برنامه کاربردی سرویس‌گیرنده، منتقل شود.

پیوستگی مشخصات امنیتی کانال ارتباطی بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت باید به وسیله یک کلاس سیاست حفاظت مسیر مشخص شود که باید مطابق تعاریف زیر، جزئیات سازوکارهای امنیتی مورد استفاده برای وقوع کانال را تعیین نماید:

کلاس‌های سیاست حفاظت مسیر

- انتها تا انتها – برای ایمن ساختن کانال بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت، باید یک کلید یا یک مجموعه کلید واحد، مورد استفاده قرار گیرد.
- بخش‌بندی‌شده – برای ایمن ساختن بخش‌های مختلف کانال بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت، باید کلیدها یا مجموعه‌های کلید متفاوت، مورد استفاده قرار گیرد.
- نامعلوم – هیچ مشخصه‌ای برای مشخصات امنیتی کانال بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت، بیان نشده است؛ قوت و ضعف مشخصات امنیتی کانال، ناملموس است.

درون یک کلاس سیاست حفاظت مسیر مشخص، باید به وسیله یک تنظیم پشته، دسته‌های متعددی از امنیت، مطابق دسته سیاست حفاظت مسیر، برقرار شود. دسته سیاست حفاظت مسیر، جنبه‌های امنیتی خاص که باید به وسیله تنظیم پشته و به طور ضمنی، سازوکارهایی که باید برای وقوع این جنبه‌ها مورد استفاده قرار گیرند را مطابق تعاریف زیر، تعیین می‌نماید

دسته‌های سیاست حفاظت مسیر

- ذاتی - نتیجه پلاتفرم + جنبه‌های پیش‌فرض کانال
- حفاظت شده - محرمانه بودن + یکپارچگی داده‌ها
- احراز هویت شده در مبدا - احراز هویت برنامه کاربردی کارت + حفاظت شده
- به طور دو طرفه احراز هویت شده- احراز هویت برنامه کاربردی کارت + احراز هویت شده در مبدا + حفاظت شده

احراز هویت برنامه کاربردی کارت و دسته‌های سیاست حفاظت مسیر احراز هویت شده در مبدا باید به وسیله درخواست CardApplicationStartSession استاندارد ملی ایران شماره۳ـ۱۶۳۸۶، واقع شوند.

ویژگی‌های امنیتی که قرار است مطابق با دسته‌های سیاست حفاظت مسیر، برقرار شوند و سازوکارهای ضمنی که ممکن است برای وقوع این جنبه‌های امنیتی، مورد استفاده قرار گیرند، به صورت زیر تعریف می‌شوند:

مشخصات و سازوکارها

- محرمانه – جلوگیری از استراق سمع در سراسر بخش کانال مشخص (سازوکارها)
  - کانال مورد اعتماد محرمانه
  - پلاتفرم اختصاصی
  - کانال اختصاصی
- یکپارچگی داده‌ها – حفظ یکپارچگی داده‌ها در سراسر بخش کانال مشخص (سازوکارها)
  - کانال مورد اعتماد MAC
  - پلاتفرم اختصاصی
  - کانال اختصاصی
- یکپارچگی مبدا – احراز هویت مربوط به هویت متمایزکننده مورد استفاده برای دسترسی به اطلاعات (سازوکارها)
  - احراز هویت برنامه کاربردی سرویس‌گیرنده (احراز هویت داخلی)
  - احراز هویت برنامه کاربردی سرویس‌گیرنده (احراز هویت خارجی)

به علت ترکیب تنظیمات گوناگون پشته، همه کلاس‌های سیاست حفاظت مسیر، بر روی همه تنظیمات، در دسترس نیستند. جدول ۱، دقیق‌ترین کلاس‌هایی که می‌تواند روی تنظیمات پشته گوناگون تعریف شده در بند ۵، به‌دست‌آیند را نشان می‌دهد.

جدول ۱- کلاس‌های سیاست حفاظت مسیر به ازای دسته بر تنظیم پشته

تنظیم پشته	حفاظت شده	مبدا سرویس‌گیرنده	احراز هویت دو طرفه
اختصاصی	انتها تا انتها	انتها تا انتها	انتها تا انتها
شبکه کامل	بخش‌بندی‌شده	بخش‌بندی‌شده	بخش‌بندی‌شده
ICC غیرشفاف	بخش‌بندی‌شده	بخش‌بندی‌شده	بخش‌بندی‌شده
اختصاصی راه دور	بخش‌بندی‌شده	بخش‌بندی‌شده	بخش‌بندی‌شده
مقیم ICC	انتها تا انتها	انتها تا انتها	بخش‌بندی‌شده
ICC راه دور	انتها تا انتها	انتها تا انتها	بخش‌بندی‌شده

در جدول ۱، "حفاظت شده" به برقراری محرمانه بودن داده‌ها (یعنی حفاظت در برابر استراق سمع‌کننده‌ها) و یکپارچگی داده‌ها (یعنی حفاظت در برابر هر تغییر در مقادیر داده‌ها) درون تنظیمات گوناگون پشته، اشاره می‌کند. "مبدا سرویس-گیرنده" به برقراری یک وضعیت احراز هویت بین برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت یا بین برنامه کاربردی کارت و برنامه کاربردی سرویس‌گیرنده، اشاره می‌نماید. "احراز هویت دو طرفه" به برقراری همزمان وضعیت‌های احراز هویت در برنامه کاربردی سرویس‌گیرنده و برنامه کاربردی کارت، اشاره دارد. هر جزء این جدول، بالاترین سطح

کلاس سیاست حفاظت مسیر را نشان می‌دهد که می‌تواند به وسیله نوع مشخصی از تنظیم پشته، برای یک دسته سیاست حفاظت مسیر مشخص، به دست آید.

برقراری یک مسیر بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت، به وسیله برنامه کاربردی سرویس-گیرنده و با انتخاب کردن یک تنظیم پشته قابل قبول که به عنوان یک جزء برگردانده شده از طریق مسیر(های) در دسترس به وسیله درخواست CardApplicationPath، در بر گرفته می‌شود و سپس اجرای یک درخواست CardApplicationconnect برای برقرار کردن یک اتصال به برنامه کاربردی کارت، حاصل می‌شود. پس از آن، مشخصات امنیتی دقیقتر، می‌تواند به وسیله اجرای یک CardApplicationStartSession با پروتکل احراز هویت مطلوب مشخص شده، برقرار گردد.

۶-۲ نگاشت ACL-ACR

تمام قاعده‌های دسترسی تعریف شده به وسیله فهرست‌های کنترل دسترسی باید به وسیله هر پیاده‌سازی لایه سازگار با استاندارد ملی ایران شماره ۳ـ۱۶۳۸۶، اعمال شود. انتظار می‌رود که چنین اعمالی، به وسیله برقرار کردن قواعد دسترسی درون برنامه کاربردی کارت با استفاده از سازوکارهای ارائه شده به وسیله یک پیاده‌سازی لایه استاندارد ملی ایران شماره ۲ـ۱۶۳۸۶انجام شود. این مانع توسعه این سازوکارهای اِعمال AR به وسیله سازوکارهای در دسترس روی ICC یا روی PICC نمی‌شود.

۶-۳ پیام‌رسانی ایمن

هر سیاست حفاظت مسیر که به پیام‌رسانی ایمن، ارجاع می‌کند باید از سازوکارهای پیوست الف، استفاده نماید. برای محاسبه متن رمزنگاشتی شده ضروری مورد نیاز سازوکارهای پیوست الف، پیام‌رسانی ایمن باید در هر پایانه از مسیر پیام-رسانی ایمن، از کلیدهای مشترک استفاده کند. پیام‌رسانی ایمن می‌تواند فقط در نقطه‌ای درون پشته، پیاده‌سازی شود که در آن تمام ارتباط با برنامه کاربردی کارت به APDUهای سازگار با استاندارد ISO/IEC 7816-4 خلاصه می‌شود. این نقطه درون پیاده‌سازی لایه استاندارد ملی ایران شماره۳ـ۱۶۳۸۶ یا درون پیاده‌سازی لایه استاندارد ملی ایران شماره ۲ـ ۱۶۳۸۶محقق می‌شود. به منظور وقوع پیام‌رسانی ایمن، کلیدهای مشترک باید در این نقطه و نیز درون برنامه کاربردی کارت، در دسترس باشند. کلید(های) مورد استفاده برای وقوع پیام‌رسانی ایمن درون یک نمونه‌سازی لایه استاندارد ملی ایران شماره۲ـ۱۶۳۸۶ ممکن است برای لایه، ذاتی باشد یا به وسیله اجزاء رویه‌ای در خلال عمل راه‌اندازی تعریف شده در استاندارد ملی ایران شماره۲ـ۱۶۳۸۶، مشتق شوند. کلید(های) مورد استفاده برای وقوع پیام‌رسانی ایمن درون یک نمونه‌سازی لایه استانداردملی ایران شماره۳ـ۱۶۳۸۶باید مستقیما به وسیله برنامه کاربردی سرویس‌گیرنده، یا از طریق اجرای موفقیت‌آمیز یک پروتکل احراز هویت مناسب فراهم شود.

هر نمونه‌سازی لایه سازگار با استاندارد ملی ایران شماره۲ـ۱۶۳۸۶ که یک قابلیت اجرای پیام‌رسانی ایمن را ارائه کند باید روی یک پلاتفرم اختصاصی قرار گیرد.

هنگامی که یک APDU به وسیله پیام‌رسانی ایمن، محافظت می‌شود، ممکن است مطابق با استاندارد ملی ایران شماره ۲ ـ ۱۶۳۸۶ به همین صورت در GCI منتقل شود یا به عنوان یک آرگومان یک **درخواست** یا **تایید** در IFD-API منتقل گردد.

۶-۴ مدیریت کلید کانال مورد اعتماد

امنیت حاصل از بکارگیری یک کانال مورد اعتماد، باید از هر پیام‌رسانی ایمن یا هر سازوکار امنیتی برنامه کاربردی به برنامه کاربردی مورد استفاده از طریق جلسه استاندارد ملی ایران شماره ۳ـ۱۶۳۸۶ بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت، مستقل باشد. سازوکارهای رمزنگاشتی کانال مورد اعتماد، شامل تمام مدیریت کلید باید از هر سازوکار هویت متمایزکننده مورد استفاده درون یک برنامه کاربردی کارت، مستقل باشد.

۷ مولفه‌های اتصال

معماری کلی تعریف شده در استاندارد ملی ایران شماره۱ـ۱۶۳۸۶تعدادی از مولفه‌های مجزا را که می‌تواند از طریق واسط-های مشخص شده به وسیله استاندارد ملی ایران شماره ۱۶۳۸۶ مورد دسترسی قرار گیرد، شناسایی می‌کند.

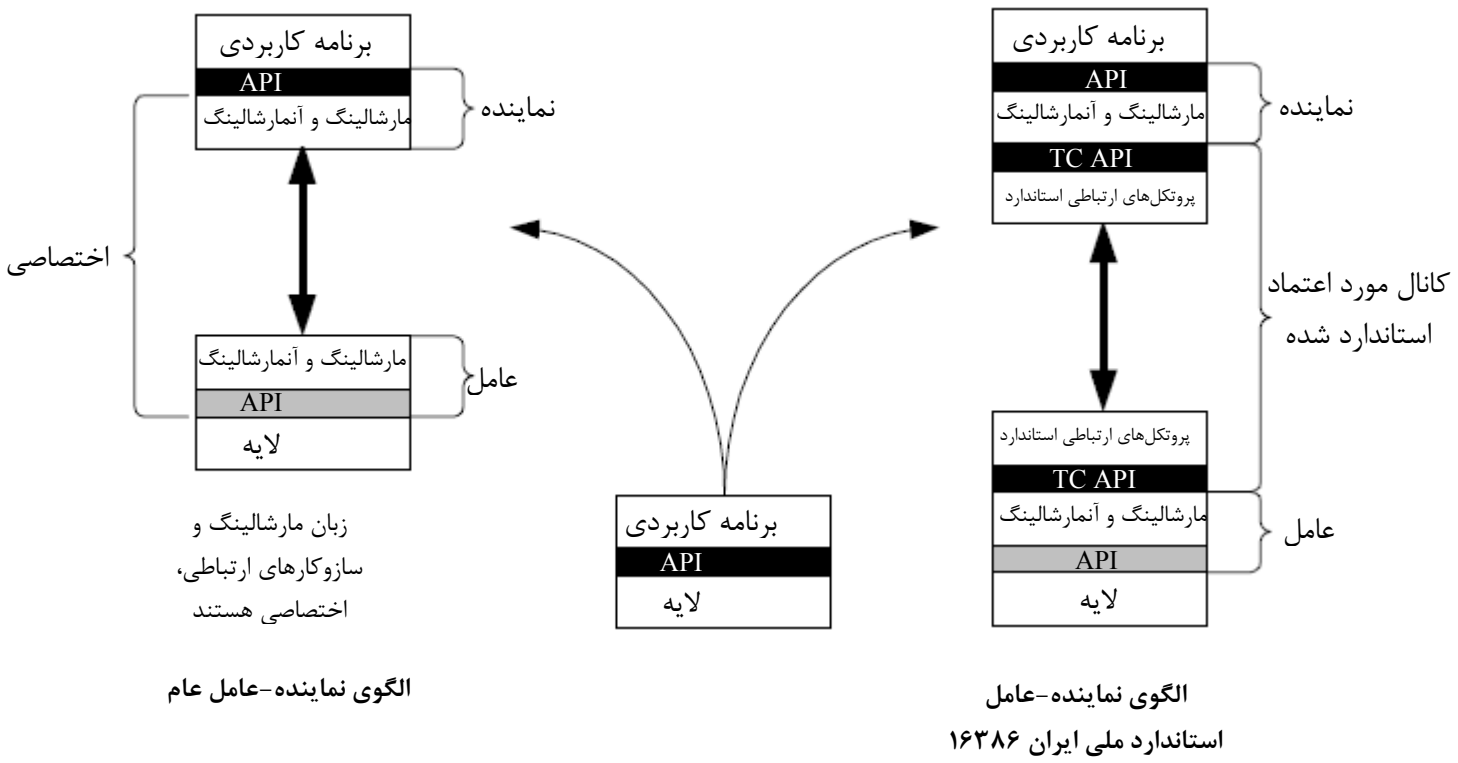
تمام واسط‌های سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ باید از طریق شماره‌های نسخه مقرر شده به وسیله مشخصات ASN.1 مربوط به واسط‌ها، به طور منحصربه‌فرد، شناسایی شود.

۷-۱ معانی درخواست و پاسخ عمل

تمام درخواست‌های پشته از API استاندارد ملی ایران شماره ۳ـ۱۶۳۸۶ به برنامه کاربردی کارت باید به طور همزمان اتفاق بیافتد.

۷-۲ معماری نماینده-عامل

استاندارد ملی ایران شماره۳ـ۱۶۳۸۶یک API قابل توسعه، API لایه دسترسی به خدمت، را تعریف می‌نماید که به وسیله آن، یک برنامه کاربردی سرویس‌گیرنده باید خدمات را از یک برنامه کاربردی کارت راه دور، به‌دست‌آورد. این API ممکن است به عنوان یک مولفه تشکیل دهنده یک نماینده، نمونه‌سازی گردد. نماینده‌ای که باید به وسیله یک اتصال خاص زبان مبتنی بر زبان پیاده‌سازی برنامه کاربردی سرویس‌گیرنده، به‌طور مستقیم به برنامه کاربردی سرویس‌گیرنده، متصل شود. به طور مشابه، IFD API تعریف شده در بند ۷ نیز می‌تواند به‌وسیله یک امکان نماینده-عامل استاندارد، به‌صورت نشان داده شده در شکل ۱۰، پیاده‌سازی شود.



شکل ۱۰- امکانات پروکسی- عامل

همانگونه که در شکل، نشان داده شده، توابع نماینده SAL-API باید باشد برای (الف) فراهم کردن یک اتصال خاص زبان مربوط بهAPI استاندارد ملی ایران شماره۳ـ۱۶۳۸۶، برای اتصال مستقیم به برنامه کاربردی سرویس‌گیرنده، (ب) مارشال کردن تمام جنبه‌های API استاندارد ملی ایران شماره۳ـ۱۶۳۸۶درون یک جریان داده‌های ASN.1، و (پ) منتقل کردن این جریان داده‌ها به پشته از طریق یک کانال مورد اعتماد. این کانال مورد اعتماد، به جریان داده‌ها امکان می‌دهد که به هر مولفه پشته دیگر، در هر مکانی روی شبکه، تحویل داده شود. از آنجایی که این تحویل، به وسیله یک کانال مورد اعتماد، انجام می‌شود، با این مولفه، امنیت کل پشته استاندارد ملی ایران شماره ۱۶۳۸۶ حفظ می‌شود. در تنظیمات پشته مناسب، امکانات مشابهی می‌تواند برای IFD API، مورد استفاده قرار گیرد.

۳-۷ واسط کانال مورد اعتماد

استاندارد ملی ایران شماره۱ـ۱۶۳۸۶یک معماری را تعریف می‌کند که مولفه‌های گوناگون پشته پروتکل استاندارد ملی ایران شماره ۱۶۳۸۶ را برای توزیع شدن در سراسر یک اتصال شبکه گسترده بین یک برنامه کاربردی سرویس‌گیرنده و یک برنامه کاربردی کارت، مد نظر قرار می‌دهد. این بند، یک API را تعریف می‌نماید که برای وقوع یک اتصال در سراسر یک شبکه عمومی، باید به وسیله مولفه‌های گوناگونی مورد استفاده قرار گیرد. این API باید دسترسی به تسهیلات ارتباطی را از طریق پروتکل‌های کانال مورد اعتماد مختلف شناسایی شده در پیوست الف-۲، فراهم نماید. یک تنظیم پشته سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ که از یک کانال مورد اعتماد برقرار شده به وسیله این API استفاده می‌کند، همان‌گونه که در استاندارد IETF RFC 2246 مشخص شده، باید حداقل از پروتکل TLS پشتیبانی کند.

جدول ۲- API کانال مورد اعتماد

تابع API	شرح وظیفه‌ای
TC_API_Open	این تابع، عملیات دست‌دهی[1] را آغاز می‌کند که به وسیله آن جهت سرویس‌گیرنده و سرویس‌دهنده بین دو انتهای کانال، برقرار می‌شود و به وسیله آن مشخصات امنیتی کانال، تعیین می‌گردد.
TC_API_Close	این تابع، کانال مورد اعتماد را پایان می‌دهد.
TC_API_Write	این تابع، از طریق کانال مورد اعتماد، یک پیام را به نقطه پایانه در انتهای دیگر کانال مورد اعتماد، منتقل می‌کند.
TC_API_Read	این تابع، یک پیام را از کانال مورد اعتماد، می‌پذیرد.
TC_API_Reset	این تابع، همه پیام‌های معوق درون کانال مورد اعتماد را پاک می‌نماید و کانال مورد اعتماد را از نو آغاز می‌کند.
TC_API_GetStatus	این تابع، وضعیت جاری کانال مورد اعتماد، شامل وضعیت تمام پیام‌های معوق درون کانال مورد اعتماد را بازیابی می‌کند.

1 - Handshake

۷-۳-۱ درخواست TC_API_Open

۷-۳-۱-۱ هدف

درخواست TC_API_Open، مرحله handshake کانال مورد اعتماد را آغاز می‌نماید.

۷-۳-۱-۲ عمل

```
OUT     status_code     TC_API_Open(
IN      octet string    remoteAddress,
IN      octet string    channelParams,
OUT     struct_handle   channelHandle
);
```

۷-۳-۱-۳ پارامترها

remoteAddress آدرس گره کانال راه دور

channelParams پارامترهای امنیتی و اتصال، شامل یک OID برای انتخاب کردن آن پروتکل خاص، به صورتی که در پیوست الف، تعریف شده است

channelHandle نگهدارنده غیرشفاف برای کانال باز شده

۷-۳-۱-۴ پیش‌نیازها

هیچ

۷-۳-۱-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_NODE_NOT_REACHABLE

۷-۳-۱-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز، یک کانال دو طرفه، به آدرس شبکه‌ای راه دور مشخص شده، برقرار می‌شود.

۷-۳-۲ درخواست TC_API_Close

۷-۳-۲-۱ هدف

درخواست TC_API_Close، کانال مورد اعتماد را پایان می‌دهد.

۷-۳-۲-۲ عمل

```
OUT     status_code     TC_API_Close(
IN      struct_handle   channelHandle
);
```

۷-۳-۲-۳ پارامترها

channelHandle نگهدارنده کانالی که در حال حاضر، باز است

۷-۳-۲-۴ پیش‌نیازها

هیچ

۷-۳-۲-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_UNKNOWN_HANDLE

۷-۳-۲-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز این درخواست، کانالی که در حال حاضر باز است، بسته خواهد شد.

۷-۳-۳ درخواست TC_API_Read

۷-۳-۳-۱ هدف

درخواست TC_API_Read، یک پیام را از کانال مورد اعتماد، بازیابی می‌نماید.

۷-۳-۳-۲ عمل

```
OUT     status_code     TC_API_Read(
IN      struct_handle   channelHandle,
OUT     octet string    message
);
```

۷-۳-۳-۳ پارامترها

channelHandle نگهدارنده غیرشفاف برای کانال باز

message حافظه میانی حاوی پیام خوانده شده از کانال

۷-۳-۳-۴ پیش‌نیازها

هیچ

۷-۳-۳-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_WARNING_BUFFER_LENGTH_EXCEEDED
API_UNKNOWN_HANDLE

۷-۳-۳-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز این درخواست، حافظه میانی "پیام"، دربرگیرنده دنباله‌ای از بایت‌های خوانده شده از کانال باز می‌باشد.

۷-۳-۴ درخواست TC_API_Write

۷-۳-۴-۱ هدف

درخواست TC_API_Write ، یک ساختار پیام را درون کانال مورد اعتماد قرار می‌دهد.

۷-۳-۴-۲ عمل

```
OUT     status_code     TC_API_Write(
IN      struct_handle   channelHandle,
IN      octet string    message
);
```

۷-۳-۴-۳ پارامترها

channelHandle نگهدارنده غیرشفاف برای کانال باز

message بافر[1] حاوی رشته‌ای از بایت‌ها

۷-۳-۴-۴ پیش‌نیازها

هیچ

۷-۳-۴-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_UNKNOWN_HANDLE

۱ـ Buffer حافظه موقتی برای ذخیره‌سازی داده‌ها به هنگام جابجایی بین دو واحد

۷-۳-۴-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز این درخواست، حافظه میانی پیام، بر روی کانال، نوشته شده است.

۷-۳-۵ درخواست TC_API_Reset

۷-۳-۵-۱ هدف

درخواست TC_API_Reset، همه پیام‌های معوق درون کانال مورد اعتماد را حذف کرده و کانال را به یک وضعیت پس از عملیات دست‌دهی، از نو آغاز می‌نماید.

۷-۳-۵-۲ عمل

```
OUT     status_code     TC_API_Reset(
IN      struct_handle   channelHandle
);
```

۷-۳-۵-۳ پارامترها

channelHandle نگهدارنده غیرشفاف برای کانال باز

۷-۳-۵-۴ پیش‌نیازها

هیچ

۷-۳-۵-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_UNKNOWN_HANDLE

۷-۳-۵-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز این درخواست، کانال در وضعیت بلافاصله پس از عملیات دست‌دهی، بدون پیام‌های معوق، قرار می‌گیرد.

۷-۳-۶ درخواست TC_API_GetStatus

۷-۳-۶-۱ هدف

درخواست TC_API_GetStatus، وضعیت جاری کانال مورد اعتماد را برمی‌گرداند.

۷-۳-۶-۲ عمل

```
OUT     status_code     TC_API_GetStatus(
IN      struct_handle   channelHandle,
OUT     octet string    statusString
);
```

۷-۳-۶-۳ پارامترها

channelHandle نگهدارنده غیرشفاف برای کانال باز

statusString وضعیت جاری کانال

۷-۳-۶-۴ پیش‌نیازها

هیچ

۷-۳-۶-۵ کدهای بازگشتی

API_OK
API_TIMEOUT_ERROR
API_UNKNOWN_ERROR
API_UNKNOWN_HANDLE

۷-۳-۶-۶ تاثیر روی وضعیت جاری

پس از اتمام موفقیت‌آمیز این درخواست، وضعیت جاری کانال باز، در پارامتر statusString، برگردانده می‌شود.

۷-۴ API مربوط به دستگاه واسط

API مربوط به دستگاه واسط، از گروه درخواست‌های زیر تشکیل می‌شود که به صورت کلی مشخص می‌شوند:

- درخواست‌های مربوط به پایانه شکاف[1]
- درخواست‌های مربوط به شکاف
- درخواست‌های مربوط به کاربر

درخواست‌های مربوط به پایانه شکاف

API مربوط به دستگاه واسط، شامل درخواست‌های مربوط به پایانه شکاف زیر می‌شود:

-EstablishContext

-ReleaseContext

-ListIFDs

-GetIFDCapabilities

-GetStatus

-Wait

-Cancel

-ControlIFD

درخواست‌های مربوط به شکاف

API مربوط به دستگاه واسط، شامل توابع مربوط به شکاف زیر می‌شود:

-Connect

-Disconnect

-BeginTransaction

-EndTransaction

-Transmit

درخواست‌های مربوط به کاربر

API مربوط به دستگاه واسط، شامل درخواست‌های مربوط به کاربر زیر می‌شود:

-VerifyUser

-ModifyVerificationData

-Output

API مربوط به دستگاه واسط تعریف شده در این بند، باید تنها از درون یک لایه SAL یا از درون یک لایه GCI، مورد دسترسی قرار گیرد. اگر کد بازگشتی هر درخواست عمل، مغایر با IFD_OK باشد، آنگاه ممکن است که پارامترهای خروجی وجود نداشته باشد یا به درستی تنظیم نشود.

---

1 - Slot Terminal

۷-۴-۱ EstablishContext

یک درخواست مربوط به پایانه شکاف.

۷-۴-۱-۱ هدف

درخواست EstablishContext، زمینه‌ای ایجاد می‌کند که از طریق آن، فرمان‌های بیشتری می‌توانند به لایه IFD ارسال شوند.

۷-۴-۱-۲ عمل

```
OUT     Status              EstablishContext (
IN      ChannelHandleType   ChannelHandle OPTIONAL,
OUT     ContextHandleType   ContextHandle
);
```

۷-۴-۱-۳ پارامترها

channelHandle — یک پارامتر اختیاری است که ممکن است برای هدایت یک کانال برقرار شده به یک سامانه راه دور، مورد استفاده قرار گیرد. ChannelHandle، با استفاده از تابع TC_API_Open تعریف شده در استاندارد ملی ایران شماره۴ـ۱۶۳۸۶، به دست می‌آید. اگر فراخوانی، به سامانه محلی هدایت شود، مجاز است که پارامتر ChannelHandle حذف گردد.

ContextHandle — نگهدارنده برگردانده شده است که ممکن است در سایر فراخوانی‌ها برای آدرس‌دهی زمینه برقرار شده با لایه IFD، مورد استفاده قرار گیرد.

۷-۴-۱-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CHANNEL_HANDLE	ChannelHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۲ ReleaseContext

یک درخواست مربوط به پایانه شکاف.

۷-۴-۲-۱ هدف

درخواست ReleaseContext، زمینه برقرار شده با لایه IFD را آزاد می‌کند.

۷-۴-۲-۲ عمل

```
OUT     Status              ReleaseContext (
IN      ContextHandleType   ContextHandle
);
```

۷-۴-۲-۳ پارامترها

ContextHandle — نگهدارنده زمینه‌ای است که باید آزاد شود.

۷-۴-۲-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۳ ListIFDs

یک درخواست مربوط به پایانه شکاف.

۷-۴-۳-۱ هدف

درخواست ListIFDs، فهرستی از دستگاه‌های واسطی (IFD) که در حال حاضر روی آن پلاتفرم، در دسترس هستند، را برمی‌گرداند.

۷-۴-۳-۲ عمل

```
OUT     Status              ListIFDs(
IN      ContextHandleType   ContextHandle,
OUT     string              IFDName[ ]
);
```

۷-۴-۳-۳ پارامترها

ContextHandle	زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.
IFDName	نام منحصربه‌فرد IFD است که برای آدرس‌دهی آن IFD بخصوص، به‌کارمی‌رود.

۷-۴-۳-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۴ GetIFDCapabilities

یک درخواست مربوط به پایانه شکاف.

۷-۴-۴-۱ هدف

درخواست GetIFDCapabilities، اطلاعاتی درباره قابلیت‌های یک IFD مشخص و واحدهای عملیاتی مربوط به آن، را برمی‌گرداند.

۷-۴-۴-۲ عمل

```
OUT     Status                  GetIFDCapabilities(
IN      ContextHandleType       ContextHandle,
IN      string                  IFDName,
OUT     IFDCapabilitiesType     IFDCapabilities
);
```

۷-۴-۴-۳ پارامترها

ContextHandle	زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.
IFDName	نام منحصربه‌فرد IFD است که برای آدرس‌دهی آن IFD بخصوص، به‌کارمی‌رود.
IFDCapabilities	شامل اطلاعاتی درباره قابلیت‌های IFD مشخص شده و واحدهای عملیاتی آن، می‌باشد. این پارامتر، از نوع IFDCapabilitiesType است که به صورت زیر، سازمان‌دهی می‌شود:

```
structure
{
SlotCapabilityType SlotCapability[ ],
DisplayCapabilityType DisplayCapability[ ],
KeyPadCapapbilityType KeyPadCapability[ ],
BioSensorCapabilityType BioSensorCapability[ ],
BooleanType OpticalSignalUnit,
```

```
BooleanType AcousticSignalUnit
} IFDCapabilitiesType
```

SlotCapability برای هر شکاف موجود IFD، وجود دارد و شامل اطلاعاتی درباره این شکاف است. این از نوع SlotCapabilityType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
nonNegativeInteger Index,
BooleanType ContactBased
} SlotCapabilityType
```

Index – اندیس شکاف است که از صفر تا تعداد شکاف‌ها منهای یک، تغییر می‌کند.

ContactBased – نشان می‌دهد که آیا شکاف برای کارت‌های مبتنی بر تماس یا برای کارت‌های بدون تماس است.

DisplayCapability برای هر صفحه نمایش IFD وجود دارد و شامل اطلاعاتی درباره قابلیت‌های آن است. این از نوع DisplayCapabilityType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
nonNegativeInteger Index,
nonNegativeInteger Lines,
nonNegativeInteger Columns,
nonNegativeInteger VirtualLines
OPTIONAL,
nonNegativeInteger VirtualColumns
OPTIONAL
}DisplayCapabilitiesType
```

Index – اندیس صفحه نمایش است که از صفر تا تعداد صفحه نمایش‌ها منهای یک، تغییر می‌کند.

Lines – حاوی تعداد خطوط قابل رویت پشتیبانی شده به وسیله صفحه نمایش پایانه کارت است.

Columns - حاوی تعداد ستون‌های قابل رویت پشتیبانی شده به وسیله صفحه نمایش پایانه کارت است.

VirtualLines – به طور اختیاری حاوی تعداد خطوط مجازی پشتیبانی شده به وسیله صفحه نمایش پایانه کارت از طریق پیمایش خطی[1] است.

VirtualColumns – به طور اختیاری حاوی تعداد ستون‌های مجازی پشتیبانی شده به وسیله صفحه نمایش پایانه کارت از طریق پیمایش سطحی[2] است.

KeyPadCapability برای هر صفحه کلید موجود IFD، وجود دارد و شامل اطلاعاتی درباره قابلیت‌های آن است. این از نوع KeyPadCapabilityType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
nonNegativeInteger Index,
positiveInteger Keys
}KeyPadCapabilitiesType
```

Index – اندیس صفحه کلید است که از صفر تا تعداد صفحه کلیدها منهای یک، تغییر می‌کند.

Keys – حاوی تعداد کلیدهای صفحه کلید است.

BioSensorCapability - برای هر حسگر زیست سنجی موجود IFD، وجود دارد و شامل اطلاعاتی درباره قابلیت‌های این حسگر زیست سنجی است. این از نوع BioSensorCapabilityType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
nonNegativeInteger Index,
```

1 - Scrolling

2 - Panning

nonNegativeInteger BiometricType
} BioSensorCapabilityType

Index – اندیس حسگر زیست سنجی است که از صفر تا تعداد حسگرهای زیست سنجی-۱ تغییر می‌کند.

BiometricType – نوع حسگر زیست سنجی را به صورتی که در استاندارد ISO/IEC 19784-1:2006, 7.8 تعریف شده، نشان می‌دهد.

OpticalSignalUnit نشان می‌دهد که آیا یک واحد علامت نوری (به عنوان مثال LED) در دسترس پایانه کارت می‌باشد.

AcousticSignalUnit نشان می‌دهد که آیا یک واحد علامت صوتی (به عنوان مثال beep) در دسترس پایانه کارت می‌باشد.

۷-۴-۴-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
API_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۵ GetStatus

یک درخواست مربوط به پایانه شکاف.

۷-۴-۵-۱ هدف

درخواست GetStatus، وضعیت جاری یک دستگاه واسط و واحدهای عملیاتی مربوط به آن، را برمی‌گرداند.

۷-۴-۵-۲ عمل

```
OUT     Status              GetStatus(
IN      ContextHandleType   ContextHandle,
IN      string              IFDName OPTIONAL,
OUT     IFDStatusType       IFDStatus [ ]
);
```

۷-۴-۵-۳ پارامترها

ContextHandle زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.

IFDName به طور اختیاری، نام یک IFD مشخص را نشان می‌دهد که باید وضعیت آن برگردانده شود. اگر این پارامتر، حذف شود وضعیت تمام IFDهای موجود، برگردانده می‌شود.

IFDStatus ممکن است برای این IFD خاص یا برای تمام IFDها وجود داشته باشد و حاوی وضعیت جاری یک IFD است. این از نوع IFDStatusType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
String IFDName,
BooleanType Connected OPTIONAL,
SlotStatusType SlotStatus [ ],
BooleanType ActiveAntenna OPTIONAL,
SimpleFUStatusType DisplayStatus [ ],
SimpleFUStatusType KeyPadStatus [ ],
SimpleFUStatusType BioSensorStatus [ ]
} IFDStatusType
```

Connected — حاوی این اطلاعات است که آیا در حال حاضر یک اتصال به IFD وجود دارد. اگر پایانه، به طور مستقیم (از طریق RS232، USB و غیره) به آن میزبان، متصل باشد ممکن است این پارامتر حذف شود.

SlotStatus — برای هر شکاف موجود IFD، وجود دارد و شامل اطلاعاتی درباره وضعیت جاری آن است. این از نوع SlotStatusType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
nonNegativeInteger Index,
BooleanType CardAvailable,
OCTET STRING ATRorATS OPTIONAL
} SlotStatusType
```

Index – اندیس شکاف است که از صفر تا تعداد شکاف‌ها منهای یک، تغییر می‌کند.

CardAvailable – اگر یک کارت در شکاف مشخص شده، قرار گرفته باشد، TRUE است.

ATRorATS – ممکن است حاوی ATR یا ATS یک کارت گرفته شده، باشد.

ActiveAntenna — حاوی وضعیت آنتن RF مورد استفاده به وسیله دستگاه(های) جفت‌شدگی مجاورتی است. اگر برای کارت‌های فاقد تماس، شکافی وجود نداشته باشد، این پارامتر ممکن است حذف گردد.

DisplayStatus — برای هر صفحه نمایش موجود IFD وجود دارد و شامل اطلاعاتی درباره وضعیت جاری آن است. این از نوع SimpleFUStatusType است که به صورت زیر سازمان‌دهی می‌شود:

```
structure
{
NonNegativeInteger Index,
BooleanType Available
} SimpleFUStatusType
```

Index – اندیس واحد عملیاتی است که از صفر تا تعداد واحدهای عملیاتی منهای یک تغییر می‌کند.

Available – نشان می‌دهد که آیا در حال حاضر، این واحد عملیاتی، اشغال است یا در دسترس درخواست‌ها قرار دارد.

KeyPadStatus — برای هر صفحه کلید موجود IFD، وجود دارد و شامل اطلاعاتی درباره وضعیت جاری آن است. این از نوع SimpleFUStatusType است که در بالا، تعریف شده است.

BioSensorStatus — برای هر حسگر زیست سنجی موجود IFD، وجود دارد و شامل اطلاعاتی درباره وضعیت جاری آن است. این از نوع SimpleFUStatusType است که در بالا، تعریف شده است.

۷-۴-۵-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۶ Wait

یک درخواست مربوط به پایانه شکاف.

۷-۴-۶-۱ هدف

درخواست Wait، به برنامه کاربردی سرویس‌گیرنده امکان می‌دهد که از وقوع یک رویداد در یک فهرست مشخص شده IFDها، اطلاع یابد.

۷-۴-۶-۲ عمل

```
OUT     Status                      Wait(
IN      ContextHandleType           ContextHandle,
IN      positiveInteger             TimeOut OPTIONAL,
IN      CallbackChannelHandleType   CallbackChannel OPTIONAL,
IN      IFDStatusType               IFDStatus [ ],
OUT     string                      SessionIdentifier OPTIONAL,
OUT     IFDStatusType               IFDEvent [ ]
);
```

۷-۴-۶-۳ پارامترها

ContextHandle	زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.
TimeOut	یک پارامتر اختیاری است که زمانی را بر حسب میلی ثانیه مشخص می‌نماید که باید برای یک رویداد منتظر ماند. اگر این پارامتر وجود نداشته باشد، این فراخوانی تا ابد، یا تا زمانی که Cancel فراخوانی شود، منتظر خواهد ماند.
CallbackChannel	یک تابع برگرداندن تماس که هنگامی که به وسیله یک رویداد از پیش تعیین شده، فعال می‌گردد اجازه یک برگرداندن تماس غیرهمزمان از تابع Wait را می‌دهد.
IFDStatus	برای هر IFD که باید بر آن نظارت شود، وجود دارد و حاوی وضعیت مفروض اخیر IFD است. این از نوع IFDStatusType است که در بالا، تعریف شده است.
SessionIdentifier	یک شناسه جلسه که جلسه Wait مشخص را هنگام فراخوانی تابع برگرداندن تماس، شناسایی می‌کند.
IFDEvent	اطلاعاتی درباره رویداد(های) به وجود آمده، برمی‌گرداند. در حالی که این نیز از نوع IFDStatusType است، بهتر است که فقط حاوی اطلاعاتی درباره رویدادهای ایجاد شده، باشد. یعنی آن زیرمجموعه‌ای از اطلاعات وضعیت که تغییر کرده است.

۷-۴-۶-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۷ Cancel

یک درخواست مربوط به پایانه شکاف.

۷-۴-۷-۱ هدف

درخواست Cancel، تلاش می‌کند که به یک فرمان اخیرا پردازش شده در یک IFD مشخص، پایان دهد.

۷-۴-۷-۲ عمل

```
OUT     Status                  Cancel (
IN      ContextHandleType       ContextHandle,
```

```
IN      string              SessionIdentifier   OPTIONAL,
IN      string              IFDName
);
```

۷-۴-۷-۳ پارامترها

ContextHandle زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.

SessionIdentifier یک شناسه جلسه که جلسه Wait مشخص در حال لغو شدن و تابع برگرداندن تماس مربوطه را شناسایی می‌کند.

IFDName نام منحصربه‌فرد IFD است که برای آدرس‌دهی آن IFD به‌کارمی‌رود و در آن، فرمان اخیرا پردازش شده، باید لغو گردد.

۷-۴-۷-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_CANCEL_NOT_POSSIBLE	عمل جاری نمی‌تواند لغو گردد.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۸ ControlIFD

یک درخواست مربوط به پایانه شکاف.

۷-۴-۸-۱ هدف

درخواست ControlIFD، امکان می‌دهد که فرمان‌ها به طور مستقیم به IFD ارسال شوند.

۷-۴-۸-۲ عمل

```
OUT     Status              ControlIFD(
IN      ContextHandleType   ContextHandle,
IN      string              IFDName,
IN      OCTET STRING        Command,
OUT     OCTET STRING        Response
);
```

۷-۴-۸-۳ پارامترها

ContextHandle زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.

IFDName نام منحصربه‌فرد IFD است که باید فرمان به آن ارسال شود.

Command فرمانی است که باید به IFD ارسال شود.

Response پاسخی است که باید به وسیله IFD برگردانده شود.

۷-۴-۸-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CHANNEL_HANDLE	ChannelHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۹ Connect

یک درخواست مربوط به شکاف.

۷-۴-۹-۱ هدف

درخواست Connect، به یک ICC در یک شکاف خاص از یک IFD متصل می‌شود.

۷-۴-۹-۲ عمل

```
OUT     Status                Connect(
IN      ContextHandleType     ContextHandle,
IN      string                IFDName,
IN      NonNegativeInteger    Slot,
IN      BooleanType           Exclusive OPTIONAL,
OUT     SlotHandleType        SlotHandle
);
```

۷-۴-۹-۳ پارامترها

ContextHandle	زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.
IFDName	نام منحصربه‌فرد IFD است که برای اتصال به یک کارت، مورد استفاده قرار می‌گیرد.
Slot	اندیس شکاف است که از صفر تا تعداد شکاف‌ها منهای یک، تغییر می‌کند.
Exclusive	نشان می‌دهد که آیا اتصال به کارت باید انحصاری یا اشتراکی باشد. اگر این پارامتر به TURE ارزیابی شود اتصال، انحصاری است. اگر این پارامتر به FALSE ارزیابی شود یا اگر این پارامتر وجود نداشته باشد، مجاز است که اتصال به کارت، با سایر درخواست‌کننده‌ها به اشتراک گذاشته شود.
SlotHandle	نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

۷-۴-۹-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_SLOT	شکاف آدرس‌دهی شده، ناشناخته است.
IFD_SHARING_VIOLATION	درخواست، موفق نبود زیرا کارت در حال حاضر به وسیله فرآیند دیگری استفاده شده است.
IFD_NO_CARD	درخواست، موفق نبود زیرا هیچ کارتی به وسیله شکاف مشخص شده، گرفته نشده است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۰ Disconnect

یک درخواست مربوط به شکاف.

۷-۴-۱۰-۱ هدف

درخواست Disconnect، اتصال به یک شکاف را پایان می‌دهد و به طور اختیاری ممکن است یک عمل اضافی، مثل بیرون دادن کارت، را نیز انجام دهد.

۷-۴-۱۰-۲ عمل

OUT	Status	**Disconnect**(
IN	SlotHandleType	SlotHandle,
IN	string	Action OPTIONAL
);		

۷-۴-۱۰-۳ پارامترها

SlotHandle نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

Action یک پارامتر اختیاری است که ممکن است یک عمل اضافی را مشخص نماید به طوری که مقادیر زیر، امکان‌پذیر هستند:

Reset	کارت را راه‌اندازی مجدد می‌کند
Unpower	کارت را خاموش می‌کند و دسترسی به آن را پایان می‌دهد
Eject	کارت را از کارت‌خوان بیرون می‌دهد
Confiscate	برای این استفاده می‌شود که نشان دهد یک کارت‌خوان تجاری پیچیده بهتر است که کارت را به ظرف مصادره[1]، انتقال دهد و آن را به کاربر برنگرداند

۷-۴-۱۰-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_ACTION	عمل درخواست شده برای اجرا، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۱ BeginTransaction

یک درخواست مربوط به شکاف.

۷-۴-۱۱-۱ هدف

درخواست BeginTransaction، یک تراکنش را آغاز می‌کند که به وسیله آن یک دنباله از درخواست‌های مرتبط می‌توانند به شکاف مشخص شده، ارسال شوند. این درخواست، دارای امکانی است که اگر هر یک از درخواست‌ها با موفقیت انجام نشوند، می‌تواند حالت قبلی را بازیابی نماید.

۷-۴-۱۱-۲ عمل

OUT	Status	**BeginTransaction**(
IN	SlotHandleType	SlotHandle
);		

۷-۴-۱۱-۳ پارامترها

SlotHandle نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

۷-۴-۱۱-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.

1 - Confiscation Bin

IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۲ EndTransaction

یک درخواست مربوط به شکاف.

۷-۴-۱۲-۱ هدف

درخواست EndTransaction، یک تراکنش که در حال حاضر با کارت مشخص شده، باز است را پایان می‌دهد.

۷-۴-۱۲-۲ عمل

```
OUT     Status          EndTransaction(
IN      SlotHandleType  SlotHandle
);
```

۷-۴-۱۲-۳ پارامترها

SlotHandle	نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

۷-۴-۱۲-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.
IFD_NO_TRANSACTION_STARTED	نشان می‌دهد که هیچ تراکنشی آغاز نشده بود.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۳ Transmit

یک درخواست مربوط به شکاف.

۷-۴-۱۳-۱ هدف

درخواست Transmit، یک APDU به کارت مشخص شده، ارسال می‌کند.

۷-۴-۱۳-۲ عمل

```
OUT     Status          Transmit(
IN      SlotHandleType  SlotHandle,
IN      OCTET STRING    InputAPDU,
OUT     OCTET STRING    OutputAPDU,
);
```

۷-۴-۱۳-۳ پارامترها

SlotHandle	نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.
InputAPDU	یک رشته بایتی حاوی یک APDU است که باید به کارت ارسال شود.
OutputAPDU	یک رشته بایتی حاوی یک APDU پاسخ از کارت است.

۷-۴-۱۳-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

VerifyUser ۱۴-۴-۷

یک درخواست مربوط به کاربر.

۱-۱۴-۴-۷ هدف

درخواست VerifyUser، تایید یک کاربر با PIN یا زیست سنجی را آغاز می‌نماید.

۲-۱۴-۴-۷ عمل

```
OUT          Status              VerifyUser(
IN           SlotHandleType      SlotHandle,
IN           InputUnitType       InputUnit,
IN           nonNegativeInteger  DisplayIndex OPTIONAL,
IN           AltVUMessagesType   AltVUMessages OPTIONAL,
IN           positiveInteger     TimeoutUntilFirstKey OPTIONAL,
IN           positiveInteger     TimeoutAfterFirstKey OPTIONAL,
IN           OCTET STRING        Template
OUT          OCTET STRING        Response
);
```

۳-۱۴-۴-۷ پارامترها

SlotHandle نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

InputUnit نشان می‌دهد که واحد ورودی باید برای گرفتن داده‌های تایید، مورد استفاده قرار گیرد. این از نوع InputUnitType است که به صورت زیر تعریف می‌شود:

```
choice
{
PinInputType PinInput,
BiometricInputType BiometricInput
} InputUnitType
```

PinInput در صورتی استفاده می‌شود که تایید کاربر باید با یک PIN (حاوی یک کلمه عبور) انجام شود. این از نوع PinInputType است که به صورت زیر تعریف می‌شود:

```
structure
{
nonNegativeInteger Index,
PasswordAttributesType
PasswordAttributes
} PinInputTypeType
```

Index اندیس صفحه‌کلیدی است که باید برای گرفتن PIN استفاده شود.

PasswordAttributes قالب PIN را به وسیله یک ساختار، مشخص می‌کند.

```
structure
{
PasswordFlagsType pwdFlagsType,
PasswordTypeType pwdType,
nonNegativeInteger minLength,
nonNegativeInteger storedLength,
nonNegativeInteger maxLength OPTIONAL,
PadCharType padChar OPTIONAL,
DateTimeType lastPasswordChange OPTIONAL,
} PasswordAttributesType
```

پارامترهای مشخص شده به وسیله ساختار PasswordAttributesType در استاندارد ISO/IEC 7816-15 تعریف شده‌اند.

BiometricInput در صورتی استفاده می‌شود که تایید کاربر با استفاده از زیست سنجی انجام شود. این از نوع BiometricInputType است که به صورت زیر تعریف می‌شود:

```
structure
```

{
nonNegativeInteger Index,
nonNegativeInteger BiometricSubType
} BiometricInputType

Index — اندیس حسگر زیست سنجی است که باید برای گرفتن داده‌های زیست سنجی، استفاده شود.

BiometricSubType — زیرنوع زیست سنجی را به صورتی که در استاندارد ISO/IEC FDIS 19784-1:2006, 7.14 تعریف شده، مشخص می‌نماید.

DisplayIndex — اندیس صفحه نمایشی است که باید برای نشان دادن پیام‌هایی برای راهنمایی کاربر، مورد استفاده قرار گیرد. اگر نباید هیچ پیامی نشان داده شود یا IFD به یک صفحه نمایش، مجهز نباشد، ممکن است این پارامتر حذف گردد.

AltVUMessages — یک پارامتر اختیاری است که برای مشخص نمودن صریح پیام‌های متفاوتی (کدگذاری شده به صورت UTF-8 مطابق با RFC3629) که باید درون فرآیند تایید، نشان داده شود، استفاده می‌گردد. این پارامتر از نوع AltVMessagesType است که به صورت زیر مشخص می‌شود:

structure
{
string AuthenticationRequestMessage OPTIONAL,
string SuccessMessage OPTIONAL,
string AuthenticationFailedMessage OPTIONAL,
string RequestConfirmationMessage OPTIONAL,
string CancelMessage OPTIONAL
} AltVUMessagesType

ممکن است پارامترهای متفاوتی برای مشخص نمودن صریح آنچه که قرار است درون فرآیند تایید نشان داده شود، مورد استفاده قرار گیرد. اگر یک پارامتر حذف شود، از یک (مجموعه) پیام(های) پیش‌فرض مناسب، استفاده خواهد شد.

TimeoutUntilFirstKey — یک پارامتر اختیاری است که زمان انتظار تا فشرده شدن اولین کلید را بر حسب میلی ثانیه، مشخص می‌کند.

TimeoutAfterFirstKey — یک پارامتر اختیاری است که زمان انتظار پس از فشرده شدن اولین کلید را بر حسب میلی ثانیه، مشخص می‌کند.

Template — الگویی است که داده‌های تایید باید به وسیله IFD، در آن درج گردد قبل از اینکه آن داده‌ها به کارت ارسال شود.

Response — پاسخ کارت است.

۷-۴-۱۴-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_INPUT_UNIT	واحد ورودی مشخص شده، ناشناخته است.
IFD_CANCELLATION_BY_USER	کاربر، عمل را لغو کرد.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.
IFD_UNKNOWN_PIN_FORMAT	قالب PIN مشخص شده، ناشناخته است.

IFD_UNKNOWN_BIOMETRIC_SUBTYPE قالب PIN مشخص شده، ناشناخته است.

۷-۴-۱۵ ModifyVerificationData

یک درخواست مربوط به کاربر.

۷-۴-۱۵-۱ هدف

درخواست ModifyVerificationData، اصلاح داده‌های تایید (الگوی PIN یا زیست سنجی) را آغاز می‌نماید. همچنین این تابع ممکن است برای رفع مسدودیت یک PIN، به وسیله فراهم کردن یک کلید رفع مسدودیت شخصی (PUK)، مورد استفاده قرار گیرد.

۷-۴-۱۵-۲ عمل

```
OUT Status ModifyVerificationData (
IN SlotHandleType SlotHandle,
IN InputUnitType InputUnit,
IN nonNegativeInteger DisplayIndex OPTIONAL,
IN AltMVDMessagesType AltMVDMessages OPTIONAL,
IN OCTET STRING OldReferenceData OPTIONAL,
IN positiveInteger TimeoutUntilFirstKey OPTIONAL,
IN positiveInteger TimeoutAfterFirstKey OPTIONAL,
IN BooleanType RepeatInput OPTIONAL,
IN OCTET STRING Template,
);
```

۷-۴-۱۵-۳ پارامترها

SlotHandle نگهدارنده‌ای است که اتصال به کارت هوشمند را شناسایی می‌نماید.

InputUnit نشان می‌دهد که کدام واحد ورودی باید برای گرفتن داده‌های تایید، مورد استفاده قرار گیرد. این از نوع InputUnitType است که در بالا تعریف شده است.

DisplayIndex اندیس صفحه نمایشی است که باید برای نشان دادن پیام‌هایی برای راهنمایی کاربر، مورد استفاده قرار گیرد. اگر نباید هیچ پیامی نشان داده شود یا IFD به یک صفحه نمایش، مجهز نباشد، ممکن است این پارامتر حذف گردد.

AltMVDMessages یک پارامتر اختیاری است که برای مشخص نمودن صریح پیام‌های متفاوتی (کدگذاری شده به صورت UTF-8 مطابق با RFC3629) که باید درون فرآیند اصلاح داده‌های تایید، نشان داده شود، استفاده می‌گردد. این پارامتر از نوع AltMVDMessagesType است که به صورت زیر مشخص می‌شود:

```
structure
{
string AuthenticationRequestMessage OPTIONAL,
string SuccessMessage OPTIONAL,
string AuthenticationFailedMessage OPTIONAL,
string EnterNewAuthenticationDataMessage OPTIONAL,
string RepeatInputMessage OPTIONAL,
string ComparisonOfRepeatedDataFailed OPTIONAL,
string RequestConfirmationMessage OPTIONAL,
string CancelMessage OPTIONAL
} AltMVDMessagesType
```

ممکن است پارامترهای متفاوتی برای مشخص نمودن صریح آنچه که قرار است درون فرآیند اصلاح داده‌های تایید نشان داده شود، مورد استفاده قرار گیرد. اگر یک پارامتر حذف شود، از یک (مجموعه) پیام(های) پیش‌فرض مناسب، استفاده خواهد شد.

OldReferenceData	یک پارامتر اختیاری است که ممکن است به وسیله برنامه کاربردی سرویس‌گیرنده برای امکان دادن به آغاز (از راه دور) اصلاح یا رفع مسدودیت داده‌های مرجع، ارائه شود. فرض می‌شود که در این مورد، هر گونه قالب‌بندی داده‌های مرجع، به وسیله برنامه کاربردی سرویس‌گیرنده انجام می‌شود. اگر داده‌های مرجع قدیمی، به وسیله IFD گرفته شود ممکن است این پارامتر حذف گردد.
TimeoutUntilFirstKey	یک پارامتر اختیاری است که زمان انتظار تا فشرده شدن اولین کلید را بر حسب میلی ثانیه، مشخص می‌کند.
TimeoutAfterFirstKey	یک پارامتر اختیاری است که زمان انتظار پس از فشرده شدن اولین کلید را بر حسب میلی ثانیه، مشخص می‌کند.
RepeatInput	نشان می‌دهد که آیا کاربر باید مجبور به تکرار ورود داده‌های تایید، شود.
Template	الگویی است که داده‌های تایید باید به وسیله IFD، در آن درج گردد قبل از اینکه آن داده‌ها درون یک فرمان CHANGE REFERENCE DATA مطابق استاندارد ISO/IEC 7816-4 به کارت ارسال شود.
Response	پاسخ کارت است.

۷-۴-۱۵-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_SLOT_HANDLE	SlotHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_INPUT_UNIT	واحد ورودی مشخص شده، ناشناخته است.
IFD_CANCELLATION_BY_USER	کاربر، عمل را لغو کرد.
IFD_REPEATED_DATA_MISMATCH	داده‌های شناسایی تکرار شده، مطابقت ندارد.
IFD_UNKNOWN_PIN_FORMAT	قالب PIN مشخص شده، ناشناخته است.
IFD_UNKNOWN_BIOMETRIC_SUBTYPE	قالب PIN مشخص شده، ناشناخته است.
IFD_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۶ Output

یک درخواست مربوط به کاربر.

۷-۴-۱۶-۱ هدف

درخواست Output، برای نمایش دادن یک پیام درIFD به‌کار می‌رود.

۷-۴-۱۶-۲ عمل

```
OUT Status Output (
IN ContextHandleType ContextHandle,
IN string IFDName,
IN OutputInfoType OutputInfo
);
```

۷-۴-۱۶-۳ پارامترها

ContextHandle	زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.
IFDName	آن IFD را مشخص می‌کند که در آن، خروجی باید قرار گیرد.

OutputInfo ساختاری که به وسیله آن، اطلاعات خروجی متغیر، منتقل می‌شود.

```
structure
{
positiveInteger Timeout OPTIONAL,
nonNegativeInteger DisplayIndex OPTIONAL,
string Message OPTIONAL,
BooleanType AcousticalSignal OPTIONAL,
BooleanType OpticalSignal OPTIONAL
} OutputInfoType
```

TimeOut یک پارامتر اختیاری است که زمانی را بر حسب میلی ثانیه مشخص می‌نماید که پیام یا علامت نمایش داده شده، ناپدید خواهد شد. اگر این پارامتر وجود نداشته باشد، آن خروجی، برای همیشه یا تا زمانی که Cancel فراخوانی شود، باقی خواهد ماند.

DisplayIndex اندیس صفحه نمایشی است که باید برای نشان دادن پیام، مورد استفاده قرار گیرد. اگر باید فقط علامت‌های نوری یا صوتی، فرستاده شود ممکن است این پارامتر حذف گردد.

Message حاوی متنی (کدگذاری شده به صورت UTF-8 مطابق با RFC3629) است که باید در صفحه نمایش مشخص شده، نشان داده شود. اگر باید فقط یک علامت نوری یا صوتی، فرستاده شود ممکن است پارامتر Message حذف گردد.

AcousticalSignal به طور اختیاری نشان می‌دهد که یک علامت صوتی باید تنظیم شود. اگر IFD، یک واحد علامت‌دهی مناسب نداشته باشد ممکن است این پارامتر، نادیده گرفته شود.

OpticalSignal به‌طور اختیاری نشان می‌دهد که یک علامت نوری باید تنظیم شود. اگر IFD، یک واحد علامت-دهی مناسب نداشته باشد ممکن است این پارامتر، نادیده گرفته شود.

۷-۴-۱۶-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.
IFD_UNKNOWN_DISPLAY_INDEX	اندیس صفحه نمایش مشخص شده، ناشناخته است.
API_UNKNOWN_ERROR	یک خطای ناشناخته وجود داشت.

۷-۴-۱۷ SignalEvent

یک تابع ارائه شده در سطح فراخوانی IFD_API.

۷-۴-۱۷-۱ هدف

یک تابع فراخوانی شده به وسیله لایه دستگاه واسط، برای علامت دادن وقوع یک IFDEvent مشخص.

۷-۴-۱۷-۲ عمل

```
OUT Status SignalEvent (
IN ContextHandleType ContextHandle,
IN string SessionIdentifier OPTIONAL,
IN IFDStatusType IFDEvent[ ],
);
```

۷-۴-۱۷-۳ پارامترها

ContextHandle زمینه برقرار شده با لایه IFD را آدرس‌دهی می‌کند.

SessionIdentifier	یک شناسه جلسه که جلسه Wait مشخص را هنگام فراخوانی تابع برگرداندن تماس، شناسایی می‌کند.
IFDEvent	اطلاعاتی درباره رویداد(های) به وجود آمده، برمی‌گرداند. در حالی که این نیز از نوع IFDStatusType است، بهتر است که فقط حاوی اطلاعاتی درباره رویدادهای ایجاد شده، باشد؛ یعنی آن زیرمجموعه‌ای از اطلاعات وضعیت که تغییر کرده است.

۷-۴-۱۷-۴ کدهای بازگشتی

IFD_OK	درخواست، موفقیت‌آمیز بود.
IFD_TIMEOUT_ERROR	قبل از اتمام، زمان درخواست به پایان رسید.
IFD_INVALID_CONTEXT_HANDLE	ContextHandle ارائه شده، نامعتبر است.
IFD_UNKNOWN_IFD	IFDName ارائه شده، ناشناخته است.

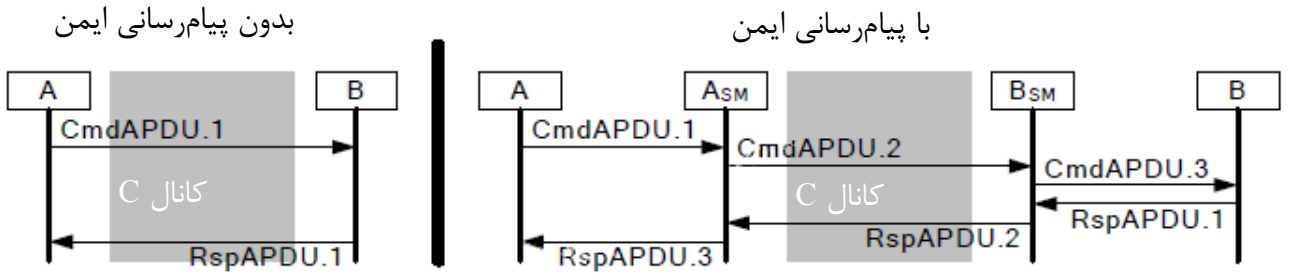
پیوست الف
(الزامی)
سازوکارهای حفاظت مسیر

الف-۱ پیام‌رسانی ایمن

این بند، فرض می‌کند که مولفه‌های A و B مطابق ۱ـ۱ـ ۱۲ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۳ـ ۷۸۱۶ از طریق کانال C، جفت‌های پاسخ فرمان را مبادله می‌کنند. این مورد در سمت چپ شکل الف-۱ نشان داده شده است. کانال C باید مطابق با ۶-۱، یک کانال مورد اعتماد با قابلیت‌های (محرمانه بودن، یکپارچگی، بدون از دست رفتن پیام، ...) شود. کانال مورد اعتماد باید به وسیله دو مولفه اضافی A_{SM} و B_{SM} روی هر سمت کانال C، به صورت نشان داده شده در سمت راست شکل الف-۱، پیاده‌سازی شود.

مولفه A_{SM} باید APDUهای فرمان دریافت شده از مولفه A را از حالت واضح[1] به حالت ایمن شده، تبدیل نماید. پس از آن، APDU فرمان ایمن شده باید از طریق کانال C به مولفه B_{SM} ارسال شود. علاوه بر این، A_{SM} باید APDUهای پاسخ را از کانال C دریافت کند، آن‌ها را به حالت واضح تبدیل می‌نماید و APDUهای پاسخ واضح را به مولفه A ارسال می‌کند.

با توجه به A_{SM}، مولفه B_{SM} باید تبدیل‌های معکوس را انجام دهد، یعنی APDUهای فرمان را غیرایمن، و APDUهای پاسخ را ایمن می‌سازد.



شکل الف-۱ – ارتباط با پیام‌رسانی ایمن و بدون آن

این بند شامل قواعد تبدیل برای موارد زیر است:

- ایمن‌سازی یک APDU فرمان در مولفه A_{SM}. یک دریافت‌کننده باید برای گرفتن APDU فرمان اصلی، تبدیل معکوس را انجام دهد.
- ایمن‌سازی یک APDU پاسخ در مولفه B_{SM}. یک دریافت‌کننده باید برای گرفتن APDU پاسخ اصلی، تبدیل معکوس را انجام دهد.

قواعد تبدیل بیان شده در اینجا، زیرمجموعه‌ای از قواعد تعریف شده در استاندارد ISO/IEC 7816-4 است. این قواعد فقط به واسطهایی اعمال می‌شود که مطابق ۱ـ۱ـ۱۲ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۳ـ ۷۸۱۶، از APDUها برای تبادل پیام، استفاده می‌گردد. در حال حاضر، درون دامنه استاندارد ملی ایران شماره ۱۶۳۸۶، کانال بین پیاده‌سازی

1 - Clear Mode

استانداردهای ملی ایران شماره۳ـ۱۶۳۸۶ و ملی ایران شماره۲ـ۱۶۳۸۶تنها کانالی است که در آن، چنین پیام‌هایی روی می‌دهد.

با توجه به سمت راست شکل الف-۱، قواعد تبدیل، بدین گونه هستند:

۱ـ CmdAPDU و CmdAPDU.3 در حالت عملیاتی بدون خطا، یکسان هستند.

۲ـ RspAPDU و RspAPDU.3 در حالت عملیاتی بدون خطا، یکسان هستند.

۳ـ محرمانه بودن CmdAPDU.1 و RspAPDU.1 تضمین می‌شود، این عمل به وسیله رمزنگاشتی داده‌های منتقل شده، انجام می‌شود.

۴ـ یکپارچگی CmdAPDU.1 و RspAPDU.1 تضمین می‌شود، این عمل با استفاده از یک MAC، انجام می‌شود.

۵ـ اصالت CmdAPDU.1 و RspAPDU.1 تضمین می‌شود، این عمل به‌وسیله همان سازوکار مورد استفاده برای یکپارچگی، انجام می‌شود.

الف-۱-۱ قواعد تبدیل اولیه

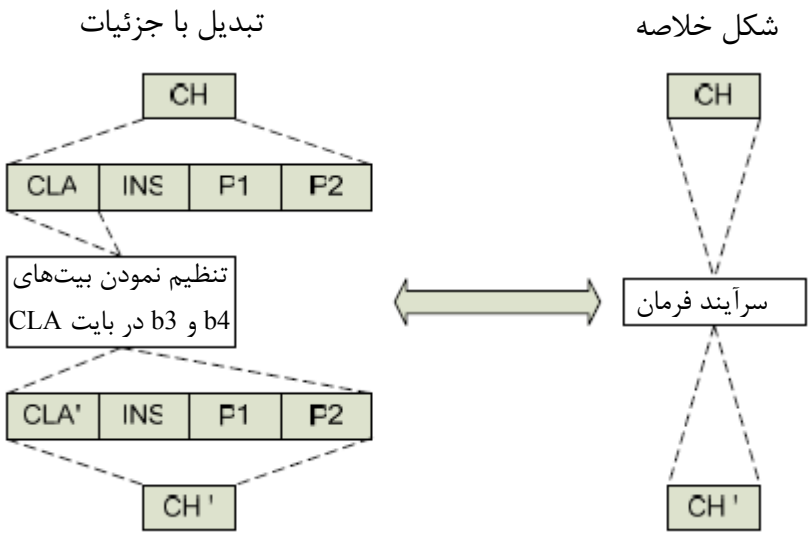
الف-۱-۱-۱ لت‌گذاری[1]

لت‌گذاری باید مطابق با بند ۶ـ۱ـ۲ استاندارد ملی ایران ـ ایزو ـ آی ای سی شماره ۱ـ۹۷۹۷(روش لت‌گذاری شماره ۲) انجام شود.

الف-۱-۱-۲ اصلاح سرآیند فرمان

یک سرآیند فرمان، CH، باید از بایت‌های CLA، INS، P1 و P2 تشکیل شود. در حین تبدیل، بیت‌های b4 و b3 در CLA باید تنظیم[2] شوند. سایر بیت‌های سرآیند نباید تغییر کنند. نتیجه، باید به وسیله ’CH نشان داده شود.

1 - Padding
2 - Set



شکل الف ـ۲ ـ تبدیل سرآیند فرمان

این بند، فرض می‌کند که بایت CLA باید مطابق با جدول ۲ از استاندارد ISO/IEC 7816-4:2005, 5.1.1، اولین مقادیر بین صنعت[1] CLA، باشد. بنابراین، شماره کانال منطقی که در CLA مشخص می‌شود باید در محدوده ۰ تا ۳ باشد. این محدودیت باید مطابق با استاندارد ملی ایران شماره۲ـ۱۶۳۸۶باشد. در صورتی که نسخه‌های بعدی این سری استانداردها، از کانال‌های منطقی با شماره‌های بزرگتر از ۳ استفاده نماید، آنگاه باید پیام‌رسانی ایمن، در بیت b6 مربوط به CLA مشخص شود و سرآیند فرمان اصلی باید در یک DO با برچسب ’89‘ کپسوله شود (رجوع کنید به جدول ۲۷ استاندارد ISO/IEC 7816-4:2005).

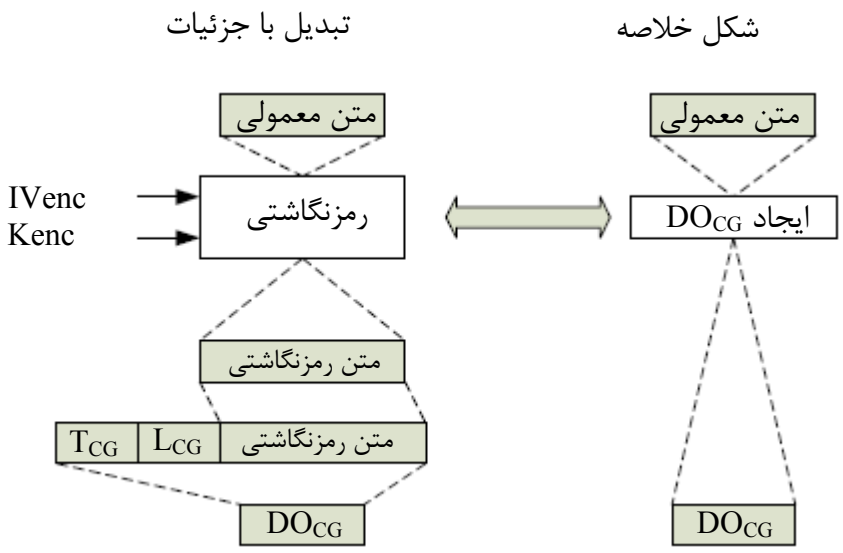
یادآوری۱ـ در GCI، فقط از کانال منطقی اولیه، پشتیبانی می‌شود.

الف-۱-۱-۳ رمزنگاشتی فیلد داده‌ها

در اینجا، نشان داده می‌شود که چگونه یک فیلد داده‌های یک APDU فرمان، برای تضمین محرمانه بودن، باید رمزنگاشتی شود. اینکه بلوک «رمزنگاشتی» چگونه کار می‌کند باید در بند الف-۱-۱-۷ مشخص شود.

یادآوری ۲ـ لت‌گذاری، بخشی از بلوک «رمزنگاشتی» است.

1 - Inter-industry

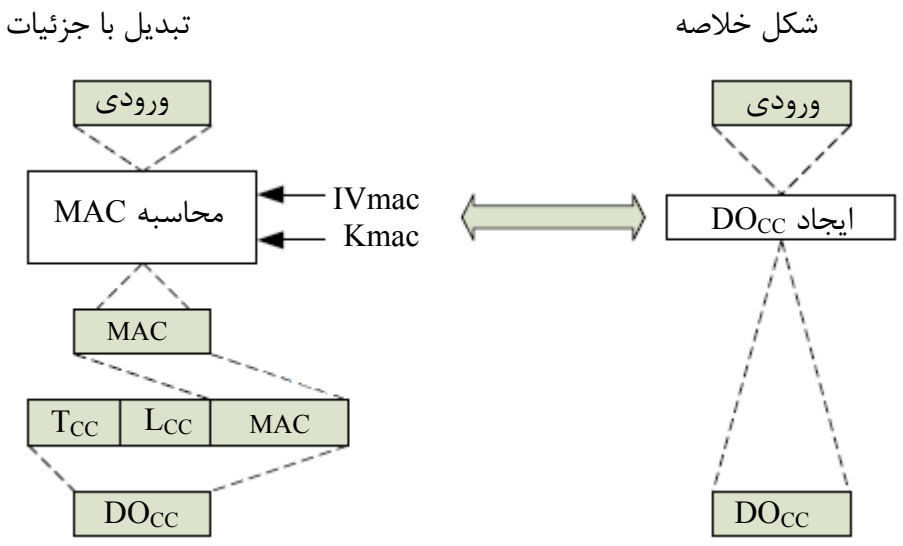


شکل الف ـ ۳ ـ ایجاد یک DO حاوی داده‌های رمزنگاشتی شده در صورت بروز کد INS فرد

الف-۱-۱-۴ محاسبه DO برای MAC

برای محاسبه MAC به بند الف-۱-۱-۶ مراجعه کنید.

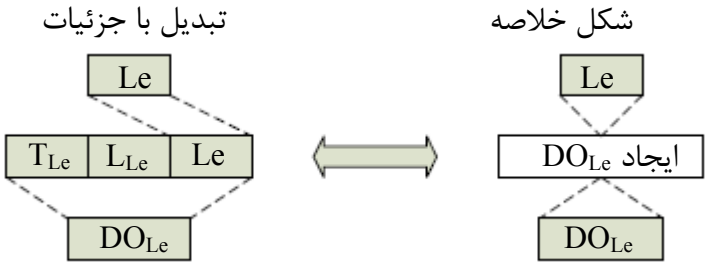
یادآوری ۳ـ لت‌گذاری، بخشی از بلوک «محاسبه MAC» است.



شکل الف ـ ۴ ـ ایجاد یک DO حاوی یک کنترل صحت داده‌های رمزنگاشتی

الف-۱-۱-۵ فیلد Le

این بند، مشخص می‌نماید که یک فیلد Le باید چگونه در یک DO، کپسوله شود.



شکل الف ـ ۵ ـ ایجاد یک DO حاوی یک فیلد Le

الف-۱-۱-۶ محاسبه MAC

این بند، شرح می‌دهد که چگونه یک کد احراز هویت باید از ورودی‌های زیر، محاسبه شود:

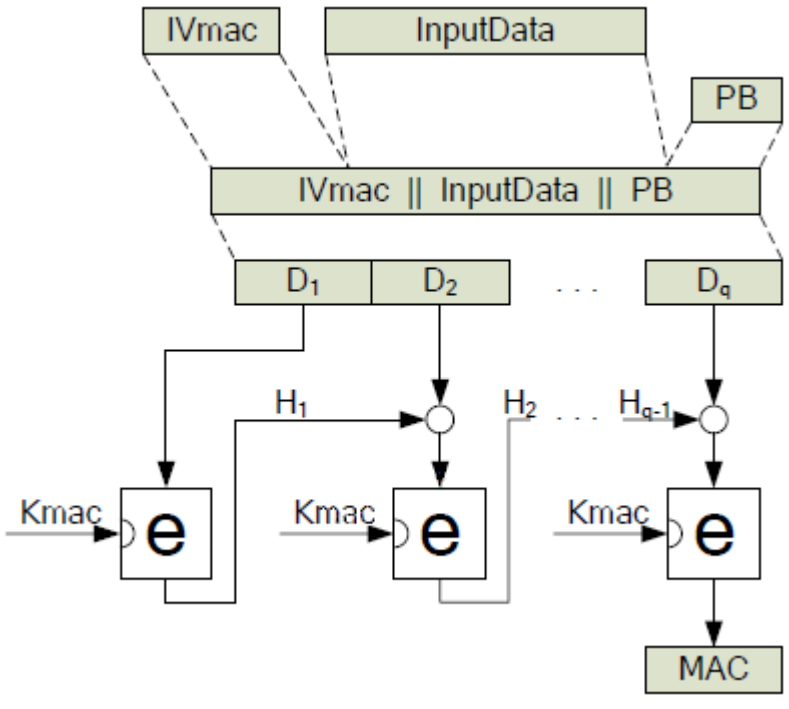
۱. IVmac یک رشته بایتی حاوی SendSequenceCounter است.

۲. InputData یک رشته بایتی اختیاری با طول اختیاری است.

۳. Kmac یک کلید مورد استفاده در الگوریتم رمزنگاشتی است.

با توجه به بند ۵ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ـ ۹۷۹۷ از موارد زیر باید استفاده شود:

- از AES به عنوان الگوریتم رمزنگاشتی بلوک، استفاده شود.
- از روش ۲ لت گذاری مربوط به بند ۶ـ۲ـ۱استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ـ ۹۷۹۷ استفاده شود.
- در شرایطی که از الحاق IVmac و InputData به عنوان رشته داده‌ای D: D = IVmac || InputData استفاده شده، از الگوریتم MAC ۱ مربوط به بند ۷ـ۱ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ـ ۹۷۹۷ استفاده گردد.
- طول MAC، m، باید ۱۲۸ بیت باشد.



شکل الف ـ ۶ ـ محاسبه MAC

الف-۱-۱-۷ رمزنگاشتی

این بند، شرح می‌دهد که یک متن رمزنگاشتی شده چگونه باید از ورودی‌های زیر، محاسبه شود:

۱ـ IVenc یک رشته بایتی حاوی SendSequenceCounter است.

۲ـ InputData یک رشته بایتی اختیاری با طول اختیاری است.

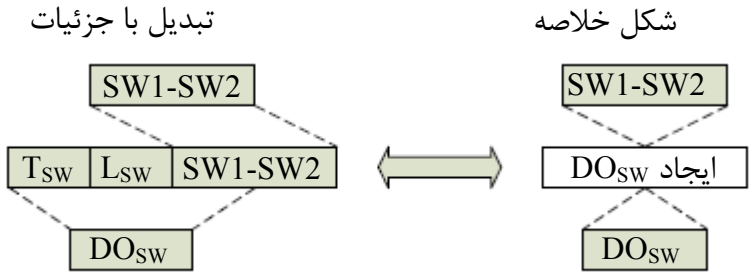
۳ـ Kenc یک کلید مورد استفاده در الگوریتم رمزنگاشتی است.

برای رمزنگاشتی از موارد زیر باید استفاده شود:

- از AES به عنوان الگوریتم رمزنگاشتی بلوک، استفاده شود.
- از روش ۲ لت گذاری مربوط به بند ۶ـ۱ـ۲ استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ـ ۹۷۹۷ برای محاسبه P = InputData || PaddingString، استفاده شود.
- برای محاسبه متن رمزنگاشتی شده از متن معمولی P، از حالت CBC (زنجیره کردن بلوک رمزنگاشتی) با مقدار اولیه IVenc، استفاده شود.

الف-۱-۱-۸ بایت‌های وضعیت

این بند، مشخص می‌نماید که بایت‌های وضعیت چگونه باید در یک DO، کپسوله شود.



شکل الف-۷ - ایجاد یک DO حاوی بایت‌های وضعیت

الف-۱-۲ ایمن‌سازی یک APDU فرمان

ابتدا، SendSequenceCounter را یک واحد، افزایش دهید. مقدار جدید SendSequenceCounter باید در مراحل ۲ و ۴، استفاده شود. سپس مراحل زیر باید انجام شود:

(۱) APDU فرمان اصلی، CmdAPDU.1، در یک DO با برچسب T_{cmd} کپسوله می‌شود. این DO در یک فرمان ENVELOPE، کپسوله می‌گردد.

(۲) سرآیند فرمان، به صورت نشان داده شده در الف-۱-۱-۲، تبدیل می‌شود و فیلد داده‌های فرمان در یک DO با برچسب T_{CG}، به صورت نشان داده شده در الف-۱-۱-۳، کپسوله می‌گردد و فیلد Le در یک DO با برچسب T_{Le}، به صورت نشان داده شده در الف-۱-۱-۵، کپسوله می‌شود.

(۳) سرآیند فرمان تبدیل شده، به صورت نشان داده شده در الف-۱-۱-۱ لت‌گذاری می‌شود. نتیجه، به DO_{CG} و DO_{Le} الحاق می‌گردد.

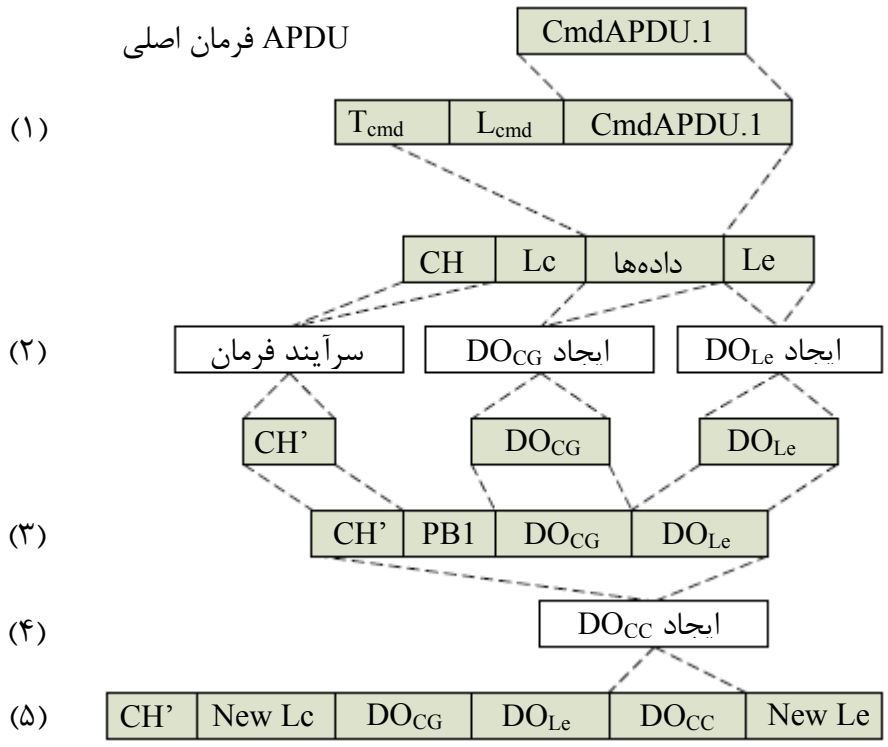
(۴) حاصل مرحله (۳)، ورودی محاسبه MAC می‌باشد.

(۵) APDU فرمان ایمن‌سازی شده، اجزای زیر را دارد:

الف- سرآیند فرمان تبدیل شده، 'CH.

ب- فیلد داده‌ای APDU فرمان ایمن‌سازی شده، حاوی DO_{CG}، DO_{Le} و DO_{CC} است.

پ- یک فیلد Le به نام «New Le».



شکل الف-۸ - ایمن‌سازی یک مورد ۴ APDU فرمان

الف-۱-۳ ایمن‌سازی یک APDU پاسخ

ابتدا، SendSequenceCounter یک واحد افزایش داده می‌شود. مقدار جدید SendSequenceCounter باید در مراحل ۲ و ۴، استفاده شود. سپس مراحل زیر باید انجام شود:

(۱) APDU پاسخ اصلی، RspAPDU.1، در یک DO با برچسب T_{rsp} کپسوله می‌شود. این DO، به عنوان فیلد داده‌های پاسخ یک APDU پاسخ، در نظر گرفته می‌شود.

(۲) داده‌های این APDU پاسخ، رمزنگاشتی می‌شود و به‌صورت نشان‌داده‌شده در الف-۱-۱-۳، در یک DO با برچسب T_{CG}، کپسوله می‌گردد و بایت‌های وضعیت، به‌صورت نشان‌داده‌شده در الف-۱-۱-۶، در یک DO با برچسب T_{SW}، کپسوله می‌شود.

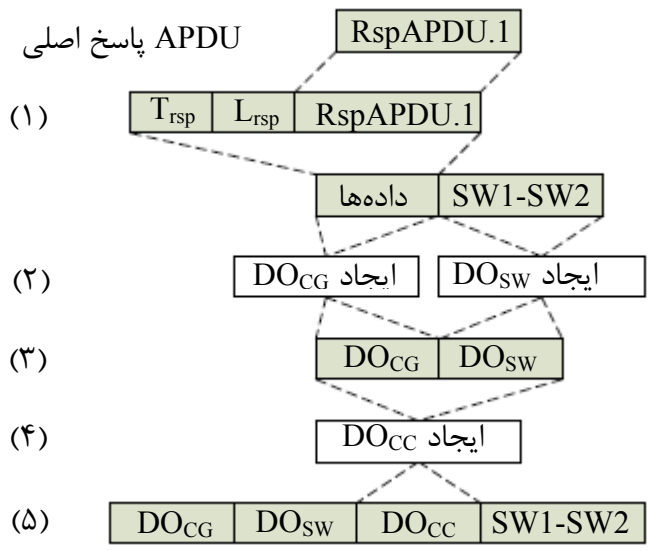
(۳) DO_{CG} و DO_{SW}، به یکدیگر الحاق می‌گردد و سپس به صورت نشان داده شده در الف-۱-۱-۱، لت‌گذاری می‌شود.

(۴) حاصل مرحله (۳)، ورودی محاسبه MAC می‌باشد.

(۵) APDU پاسخ ایمن‌سازی شده، حاوی اجزاء زیر است:

الف- فیلد داده‌ای، نتیجه الحاق DO_{CG}، DO_{SW} و DO_{CC} به یکدیگر است.

ب- دنباله، حاوی بایت‌های وضعیت «SW1-SW2» است.



شکل الف-۹ - ایمن‌سازی یک APDU پاسخ با یک فیلد داده‌ای

الف-۱-۴ تعاریف

جدول الف-۱ مقادیر مورد استفاده در بند الف-۱

اصطلاح	مقدار
CH	CLA INS P1 P2 = ’00 C3 0000’
Le	‘0000’
New Le	‘0000’
SW1-SW2	’90 00’
T_{cmd}	‘52’ = فرمان برای اجرا
T_{rsp}	‘53’ = داده‌های اختیاری
T_{CG}	‘85’
T_{Le}	‘97’
T_{CC}	‘8E’
T_{SW}	‘99’
SendSequenceCounter	رشته بایتی، حاصل برقراری یک کانال مورد اعتماد
Kenc	کلید، حاصل برقراری یک کانال مورد اعتماد
Kmac	کلید، حاصل برقراری یک کانال مورد اعتماد

الف-۲ پروتکل‌های مسیر مورد اعتماد

یک پشته میان‌افزار سازگار با استاندارد ملی ایران شماره ۱۶۳۸۶ باید یک پروتکل TLS را در پیاده‌سازی کانال مورد اعتماد بکار گرفته شده برای اتصال مولفه‌های پشته گوناگون، بجز اتصال به ICC، فراهم نماید. ممکن است پروتکل‌های اضافی به صورت توصیف شده در بندهای بعدی این پیوست، فراهم گردد. پیام‌رسانی ایمن تعریف شده در استاندارد ISO/IEC 7816-4 مجاز است که برای اتصال به ICC مورد استفاده قرار گیرد.

الف-۲-۱ امنیت لایه انتقال (TLS)

پروتکل به صورت مشخص شده در: IETF RFC 2246، پروتکل TLS نسخه ۱.۰.

پیوست ب

(الزامی)

IFD-API: اتصال خدمت وب

اتصال خدمت وب برای IFD-API در فایل‌های زیر، تعریف شده است:

- ISOCommon.XSD – نوع‌های اولیه مانند ResponseType را تعریف می‌کند که اساس تمام پیام‌های پاسخ را شکل می‌دهد.
- ISOIFD.XSD – که در آن، پارامترهای هر درخواست و پاسخ IFD-API به صورت اجزاء XML مشخص شده‌اند.
- ISOIFD.WSDL – که دربرگیرنده ISOIFD.XSD است و اتصال خدمت وب را برای IFD-API مشخص می‌نماید.

ب-۱ ویژگی ISOCommon.XSD

```
 <?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:dss="urn:oasis:names:tc:dss:1.0:core:schema"
version="1.1">
<import namespace="urn:oasis:names:tc:dss:1.0:core:schema"
schemaLocation="oasis-dss-core-schema-v1.0-os.xsd"></import>
<!--
<element name="COMMONSchemaVersion">
<complexType attribute name="schemaVersion" type="decimal" use="required"/>
</element>
<!-- Definition of Basic Types -->
<simpleType name="SlotHandleType">
</simpleType>
<complexType name="ChannelHandleType">
<sequence>
<element name="ProtocolTerminationPoint" type="anyURI" maxOccurs="1"
minOccurs="0"></element>
<element name="SessionIdentifier" type="string" maxOccurs="1"
minOccurs="0"></element>
<element name="Binding" type="anyURI" maxOccurs="1"
minOccurs="0" default="http://schemas.xmlsoap.org/soap/http">
</element>
</sequence>
</complexType>
<simpleType name="ContextHandleType">
<restriction base="hexBinary">
</restriction>
</simpleType>
<!-- Define Response Type -->
<complexType name="RequestType">
<complexContent>
<restriction base="dss:RequestBaseType">
</restriction>
</complexContent>
</complexType>
<complexType name="ResponseType">
<complexContent>
<restriction base="dss:ResponseBaseType">
<sequence>
<element ref="dss:Result"/>
</sequence>
</restriction>
```

</complexContent>
</complexType>
</schema>

ب-۲ مشخصه ISOIFD.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
version="1.1">
<!--
<!-- Definition of Basic Types -->
<include schemaLocation="ISOCommon.xsd"></include>
<element name="IFDSchemaVersion">
<complexType attribute name="schemaVersion" type="decimal" use="required"/>
</element>
<!-- Card terminal related functions -->
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ChannelHandle"
type="iso:ChannelHandleType" maxOccurs="1" minOccurs="0"
/>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="EstablishContextResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="0">
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- ReleaseContext -->
<element name="ReleaseContext">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="ReleaseContextResponse" type="iso:ResponseType">
</element>
<!-- ListIFDs -->
<element name="ListIFDs">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
```

```
<element name="ContextHandle"
type="iso:ContextHandleType" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="ListIFDsResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="IFDName" maxOccurs="unbounded"
minOccurs="0" type="string" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- GetIFDCapabilities -->
<element name="GetIFDCapabilities">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" />
<element name="IFDName" type="string" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="GetIFDCapabilitiesResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence maxOccurs="1" minOccurs="0">
<element name="IFDCapabilities" maxOccurs="1"
minOccurs="1" type="iso:IFDCapabilitiesType" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<complexType name="IFDCapabilitiesType">
<sequence>
<element name="SlotCapability" type="iso:SlotCapabilityType"
maxOccurs="unbounded" minOccurs="1"/>
<element name="DisplayCapability" type="iso:DisplayCapabilityType"
maxOccurs="unbounded" minOccurs="0"/>
<element name="KeyPadCapability" type="iso:KeyPadCapabilityType"
maxOccurs="unbounded" minOccurs="0"></element>
<element name="BioSensorCapability" type="iso:BioSensorCapabilityType"
maxOccurs="unbounded" minOccurs="0"></element>
<element name="OpticalSignalUnit" type="boolean"></element>
<element name="AcousticSignalUnit" type="boolean"></element>
</sequence>
</complexType>
<complexType name="SlotCapabilityType">
<sequence>
<element name="Index" type="nonNegativeInteger" maxOccurs="1"
minOccurs="1"/>
<element name="ContactBased" type="boolean"></element>
```

```
</sequence>
</complexType>
<complexType name="DisplayCapabilityType">
<sequence>
<element name="Index" type="nonNegativeInteger"
maxOccurs="1" minOccurs="1" />
<element minOccurs="1" maxOccurs="1" name="Lines"
type="nonNegativeInteger" />
<element name="Columns" type="nonNegativeInteger" />
<element name="VirtualLines" type="nonNegativeInteger"
maxOccurs="1" minOccurs="0" />
<element name="VirtualColumns" type="nonNegativeInteger"
maxOccurs="1" minOccurs="0" />
</sequence>
</complexType>
<complexType name="KeyPadCapabilityType">
<sequence>
<element name="Index" type="nonNegativeInteger"
maxOccurs="1" minOccurs="1" />
<element minOccurs="1" maxOccurs="1" name="Keys"
type="positiveInteger" />
</sequence>
</complexType>
<complexType name="BioSensorCapabilityType">
<sequence>
<element name="Index" type="nonNegativeInteger"
maxOccurs="1" minOccurs="1" />
<element minOccurs="1" maxOccurs="1" name="BiometricType"
type="nonNegativeInteger" />
</sequence>
</complexType>
<!-- GetStatus -->
<element name="GetStatus">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="IFDName" type="string"
maxOccurs="1" minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="GetStatusResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence maxOccurs="1" minOccurs="1">
<element name="IFDStatus" maxOccurs="unbounded"
minOccurs="0" type="iso:IFDStatusType" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<complexType name="IFDStatusType">
<sequence>
<element name="IFDName" type="string" maxOccurs="1"
minOccurs="0"></element>
<element name="Connected" type="boolean" maxOccurs="1"
```

```
minOccurs="0" />
<element minOccurs="1" maxOccurs="unbounded"
name="SlotStatus" type="iso:SlotStatusType">
<annotation>
<documentation>Index of the slot.</documentation>
</annotation>
</element>
<element name="ActiveAntenna" type="boolean" maxOccurs="1"
minOccurs="0" />
<element minOccurs="0" maxOccurs="unbounded" name="DisplayStatus"
type="iso:SimpleFUStatusType">
<annotation>
<documentation>Index of the display.</documentation>
</annotation>
</element>
<element minOccurs="0" maxOccurs="unbounded" name="KeyPadStatus"
type="iso:SimpleFUStatusType">
<annotation>
<documentation>Index of the keypad.</documentation>
</annotation>
</element>
<element minOccurs="0" maxOccurs="unbounded"
name="BioSensorStatus" type="iso:SimpleFUStatusType" />
</sequence>
</complexType>
<complexType name="SlotStatusType">
<sequence>
<element name="Index" type="nonNegativeInteger"
maxOccurs="1" minOccurs="1" />
<element minOccurs="1" maxOccurs="1" name="CardAvailable"
type="boolean" />
<element name="ATRorATS" type="hexBinary" maxOccurs="1"
minOccurs="0"></element>
</sequence>
</complexType>
<complexType name="SimpleFUStatusType">
<sequence>
<element name="Index" type="nonNegativeInteger" />
<element name="Available" type="boolean" />
</sequence>
</complexType>
<!-- Wait -->
<element name="Wait">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="TimeOut" type="positiveInteger"
maxOccurs="1" minOccurs="0" />
<element name="IFDStatus"
type="iso:IFDStatusType" maxOccurs="unbounded"
minOccurs="0" />
<element name="Callback"
type="iso:ChannelHandleType" maxOccurs="1" minOccurs="0">
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="WaitResponse">
```

```
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence maxOccurs="1" minOccurs="1">
<element name="IFDEvent" type="iso:IFDStatusType"
maxOccurs="unbounded" minOccurs="0"></element>
<element name="SessionIdentifier" type="string" maxOccurs="1"
minOccurs="0"></element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- Cancel -->
<element name="Cancel">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="IFDName" type="string" maxOccurs="1"
minOccurs="0"/><element
name="SessionIdentifier" type="string" maxOccurs="1"
minOccurs="0">
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="CancelResponse" type="iso:ResponseType" />
<!-- ControlIFD -->
<element name="ControlIFD">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="IFDName" type="string" />
<element name="Command" type="hexBinary" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="ControlIFDResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="Response" type="hexBinary" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- Card related functions -->
<!-- Connect -->
<element name="Connect">
<complexType>
```

```
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="IFDName" type="string" />
<element name="Slot" type="nonNegativeInteger" />
<element name="Exclusive" type="boolean"
maxOccurs="1" minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="ConnectResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType">
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- Disconnect -->
<element name="Disconnect">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType" />
<element name="Action" type="iso:ActionType"
maxOccurs="1" minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="DisconnectResponse" type="iso:ResponseType" />
<simpleType name="ActionType">
<restriction base="string">
<enumeration value="Reset" />
<enumeration value="Unpower" />
<enumeration value="Eject" />
<enumeration value="Confiscate" />
</restriction>
</simpleType>
<!-- BeginTransaction -->
<element name="BeginTransaction">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType" />
</sequence>
</extension>
</complexContent>
</complexType>
```

```
</element>
<element name="BeginTransactionResponse" type="iso:ResponseType" />
<!-- EndTransaction -->
<element name="EndTransaction">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="EndTransactionResponse" type="iso:ResponseType" />
<!-- Transmit -->
<element name="Transmit">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType" />
<element name="InputAPDU" type="hexBinary" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="TransmitResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="OutputAPDU" type="hexBinary"></element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!--User related functions -->
<!-- VerifyUser -->
<element name="VerifyUser">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
type="iso:SlotHandleType" />
<element name="InputUnit"
type="iso:InputUnitType" />
<element name="DisplayIndex"
type="nonNegativeInteger" maxOccurs="1" minOccurs="0">
</element>
<element name="AltVUMessages"
type="iso:AltVUMessagesType" maxOccurs="1" minOccurs="0"
/>
<element name="TimeoutUntilFirstKey"
type="positiveInteger" maxOccurs="1" minOccurs="0" />
<element name="TimeoutAfterFirstKey"
type="positiveInteger" maxOccurs="1" minOccurs="0" />
<element name="Template" type="hexBinary" />
</sequence>
```

```
</extension>
</complexContent>
</complexType>
</element>
<element name="VerifyUserResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="Response" type="hexBinary"
maxOccurs="1" minOccurs="1" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<complexType name="InputUnitType">
<choice>
<element name="PinInput" type="iso:PinInputType"></element>
<element name="BiometricInput"
type="iso:BiometricInputType">
</element>
</choice>
</complexType>
<complexType name="PinInputType">
<sequence>
<element name="Index" type="nonNegativeInteger" />
<element name="PasswordAttributes" type="iso:PasswordAttributesType"
maxOccurs="1" minOccurs="0"/>
</sequence>
</complexType>
<simpleType name="PadCharType">
<restriction base="hexBinary">
<length value="1" fixed="true"/>
</restriction>
</simpleType>
<complexType name="PasswordAttributesType">
<sequence>
<element name="pwdFlags"
type="iso:PasswordFlagsType">
</element>
<element name="pwdType"
type="iso:PasswordTypeType">
</element>
<element name="minLength"
type="nonNegativeInteger">
</element>
<element name="storedLength"
type="nonNegativeInteger">
</element>
<element name="maxLength"
type="nonNegativeInteger" maxOccurs="1" minOccurs="0">
</element>
<element name="padChar" type="iso:PadCharType"
maxOccurs="1" minOccurs="0">
</element>
<element name="lastPasswordChange"
type="dateTime" maxOccurs="1" minOccurs="0">
</element>
</sequence>
</complexType>
<simpleType name="PasswordFlagsType">
<union memberTypes="iso:BitString">
<simpleType>
```

```
<list>
<simpleType>
<restriction base="token">
<enumeration value="case-sensitive" />
<enumeration value="local" />
<enumeration value="change-disabled" />
<enumeration value="unblock-disabled" />
<enumeration value="initialized" />
<enumeration value="needs-padding" />
<enumeration value="unblockingPassword" />
<enumeration value="soPassword" />
<enumeration value="disable-allowed" />
<enumeration value="integrity-protected" />
<enumeration value="confidentiality-protected" />
<enumeration value="exchangeRefData" />
<enumeration value="resetRetryCounter1" />
<enumeration value="resetRetryCounter2" />
</restriction>
</simpleType>
</list>
</simpleType>
</union>
</simpleType>
<simpleType name="PasswordTypeType">
<restriction base="string">
<enumeration value="bcd" />
<enumeration value="ascii-numeric" />
<enumeration value="utf8" />
<enumeration value="half-nibble-bcd" />
<enumeration value="iso9564-1" />
</restriction>
</simpleType>
<simpleType name="BitString">
<restriction base="string">
<pattern value="[0-1]{0,}" />
</restriction>
</simpleType>
<complexType name="BiometricInputType">
<sequence>
<element name="Index" type="nonNegativeInteger" />
<element name="BiometricSubtype" type="nonNegativeInteger" />
</sequence>
</complexType>
<complexType name="AltVUMessagesType">
<sequence>
<element name="AuthenticationRequestMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="SuccessMessage" type="string" maxOccurs="1"
minOccurs="0" />
<element name="AuthenticationFailedMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="RequestConfirmationMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="CancelMessage" type="string" maxOccurs="1"
minOccurs="0" />
</sequence>
</complexType>
<!--ModifyVerificationData -->
<element name="ModifyVerificationData">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SlotHandle"
```

```
type="iso:SlotHandleType" maxOccurs="1" minOccurs="1" />
<element name="InputUnit"
type="iso:InputUnitType" maxOccurs="1" minOccurs="1" />
<element name="DisplayIndex"
type="nonNegativeInteger" maxOccurs="1" minOccurs="0">
</element>
<element name="AltMVDMessages"
type="iso:AltMVDMessagesType" maxOccurs="1"
minOccurs="0">
</element>
<element name="OldReferenceData"
type="hexBinary" maxOccurs="1" minOccurs="0" />
<element name="TimeoutUntilFirstKey"
type="positiveInteger" maxOccurs="1" minOccurs="0">
</element>
<element name="TimeoutAfterFirstKey"
type="positiveInteger" maxOccurs="1" minOccurs="0">
</element>
<element name="RepeatInput" type="boolean"
maxOccurs="1" minOccurs="0" />
<element name="Template" type="hexBinary"
maxOccurs="1" minOccurs="1" />
</sequence>
</extension>
</complexContent>
</complexType>
        </element>
<complexType name="AltMVDMessagesType">
<sequence>
<element name="AuthenticationRequestMessage " type="string"
maxOccurs="1" minOccurs="0" />
<element name="SuccessMessage" type="string" maxOccurs="1"
minOccurs="0" />
<element name="AuthenticationFailedMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="EnterNewAuthenticationDataMessage"
type="string" maxOccurs="1" minOccurs="0" />
<element name="RepeatInputMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="ComparisonOfRepeatedDataFailed" type="string"
maxOccurs="1" minOccurs="0" />
<element name="RequestConfirmationMessage" type="string"
maxOccurs="1" minOccurs="0" />
<element name="CancelMessage" type="string" maxOccurs="1"
minOccurs="0" />
</sequence>
</complexType>
<element name="ModifyVerificationDataResponse">
<complexType>
<complexContent>
<extension base="iso:ResponseType">
<sequence>
<element name="Response"
type="hexBinary" maxOccurs="1" minOccurs="1">
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<!-- Output -->
<element name="Output">
<complexType>
<complexContent>
```

```
<extension base="iso:OutputType">
<sequence>
<element name="ContextHandle"
type="iso:ContextHandleType" maxOccurs="1" minOccurs="1"
/>
<element name="IFDName" type="string" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<complexType name="OutputType">
<sequence>
<element name="Timeout" type="positiveInteger" maxOccurs="1"
minOccurs="0">
</element>
<element name="DisplayIndex" type="nonNegativeInteger"
maxOccurs="1" minOccurs="0">
</element>
<element name="Message" type="string"
maxOccurs="1" minOccurs="0" />
<element name="AcousticalSignal" type="boolean"
maxOccurs="1" minOccurs="0">
</element>
<element name="OpticalSignal" type="boolean"
maxOccurs="1" minOccurs="0">
</element>
</sequence>
</complexType>
<element name="OutputResponse" type="iso:ResponseType" />
</schema>
```

ب-۳ مشخصه ISOIFD.WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap=http://schemas.xmlsoap.org/wsdl/soap/
version="1.1">
<!--
<!-- ================== -->
<!-- Definition of types -->
<!-- (only include XSDs) -->
<!-- ================== -->
<wsdl:types>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
targetNamespace="urn:iso:std:iso-iec:24727:tech:schema">
<xsd:include schemaLocation="ISOIFD.xsd" />
</xsd:schema>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
</xsd:schema>
</wsdl:types>
<!-- ===================== -->
<!-- Definition of messages -->
<!-- ===================== -->
<!-- ============================== -->
<!-- Card terminal related functions -->
<!-- (messages) -->
<!-- ============================== -->
<!-- EstablishContext -->
<wsdl:message name="EstablishContext">
<wsdl:part name="parameters" element="iso:EstablishContext" />
```

```
</wsdl:message>
<wsdl:message name="EstablishContextResponse">
<wsdl:part name="parameters" element="iso:EstablishContextResponse" />
</wsdl:message>
<!-- ReleaseContext -->
<wsdl:message name="ReleaseContext">
<wsdl:part name="parameters" element="iso:ReleaseContext" />
</wsdl:message>
<wsdl:message name="ReleaseContextResponse">
<wsdl:part name="parameters" element="iso:ReleaseContextResponse" />
</wsdl:message>
<!-- ListIFDs -->
<wsdl:message name="ListIFDs">
<wsdl:part name="parameters" element="iso:ListIFDs" />
</wsdl:message>
<wsdl:message name="ListIFDsResponse">
<wsdl:part name="parameters" element="iso:ListIFDsResponse" />
</wsdl:message>
<!-- GetIFDCapabilities -->
<wsdl:message name="GetIFDCapabilities">
<wsdl:part name="parameters" element="iso:GetIFDCapabilities" />
</wsdl:message>
<wsdl:message name="GetIFDCapabilitiesResponse">
<wsdl:part name="parameters" element="iso:GetIFDCapabilitiesResponse" />
</wsdl:message>
<!-- GetStatus -->
<wsdl:message name="GetStatus">
<wsdl:part name="parameters" element="iso:GetStatus" />
</wsdl:message>
<wsdl:message name="GetStatusResponse">
<wsdl:part name="parameters" element="iso:GetStatusResponse" />
</wsdl:message>
<!-- Wait -->
<wsdl:message name="Wait">
<wsdl:part name="parameters" element="iso:Wait" />
</wsdl:message>
<wsdl:message name="WaitResponse">
<wsdl:part name="parameters" element="iso:WaitResponse" />
</wsdl:message>
<!-- Cancel -->
<wsdl:message name="Cancel">
<wsdl:part name="parameters" element="iso:Cancel" />
</wsdl:message>
<wsdl:message name="CancelResponse">
<wsdl:part name="parameters" element="iso:CancelResponse" />
</wsdl:message>
<!-- ControlIFD -->
<wsdl:message name="ControlIFD">
<wsdl:part name="parameters" element="iso:ControlIFD" />
</wsdl:message>
<wsdl:message name="ControlIFDResponse">
<wsdl:part name="parameters" element="iso:ControlIFD" />
</wsdl:message>
<!-- ====================== -->
<!-- Card related functions -->
<!-- (messages) -->
<!-- ====================== -->
<!-- Connect -->
<wsdl:message name="Connect">
<wsdl:part name="parameters" element="iso:Connect" />
</wsdl:message>
<wsdl:message name="ConnectResponse">
<wsdl:part name="parameters" element="iso:ConnectResponse" />
</wsdl:message>
```

```
<!-- Disconnect -->
<wsdl:message name="Disconnect">
<wsdl:part name="parameters" element="iso:Disconnect" />
</wsdl:message>
<wsdl:message name="DisconnectResponse">
<wsdl:part name="parameters" element="iso:DisconnectResponse" />
</wsdl:message>
<!-- BeginTransaction -->
<wsdl:message name="BeginTransaction">
<wsdl:part name="BeginTransaction" element="iso:BeginTransaction" />
</wsdl:message>
<wsdl:message name="BeginTransactionResponse">
<wsdl:part name="BeginTransactionResponse"
element="iso:BeginTransactionResponse" />
</wsdl:message>
<!-- EndTransaction -->
<wsdl:message name="EndTransaction">
<wsdl:part name="parameters" element="iso:EndTransaction" />
</wsdl:message>
<wsdl:message name="EndTransactionResponse">
<wsdl:part name="parameters" element="iso:EndTransactionResponse" />
</wsdl:message>
<!-- Transmit -->
<wsdl:message name="Transmit">
<wsdl:part name="parameters" element="iso:Transmit" />
</wsdl:message>
<wsdl:message name="TransmitResponse">
<wsdl:part name="parameters" element="iso:TransmitResponse" />
</wsdl:message>
<!-- ====================== -->
<!-- User related functions -->
<!-- (messages) -->
<!-- ====================== -->
<!-- VerifyUser -->
<wsdl:message name="VerifyUser">
<wsdl:part name="parameters" element="iso:VerifyUser" />
</wsdl:message>
<wsdl:message name="VerifyUserResponse">
<wsdl:part name="parameters" element="iso:VerifyUserResponse" />
</wsdl:message>
<!-- ModifyVerificationData -->
<wsdl:message name="ModifyVerificationData">
<wsdl:part name="parameters" element="iso:ModifyVerificationData" />
</wsdl:message>
<wsdl:message name="ModifyVerificationDataResponse">
<wsdl:part name="parameters" element="iso:ModifyVerificationDataResponse"
/>
</wsdl:message>
<!-- Output -->
<wsdl:message name="Output">
<wsdl:part name="parameters" element="iso:Output" />
</wsdl:message>
<wsdl:message name="OutputResponse">
<wsdl:part name="parameters" element="iso:OutputResponse" />
</wsdl:message>
<!-- ====================== -->
<!-- Definition of portType -->
<!-- ====================== -->
<wsdl:portType name="IFD">
<!-- ============================== -->
<!-- Card terminal related functions -->
<!-- (portType) -->
<!-- ============================== -->
<!-- EstablishContext -->
```

```
<wsdl:operation name="EstablishContext">
<wsdl:input message="iso:EstablishContext" />
<wsdl:output message="iso:EstablishContextResponse" />
</wsdl:operation>
<!-- ReleaseContext -->
<wsdl:operation name="ReleaseContext">
<wsdl:input message="iso:ReleaseContext" />
<wsdl:output message="iso:ReleaseContextResponse" />
</wsdl:operation>
<!-- ListIFDs -->
<wsdl:operation name="ListIFDs">
<wsdl:input message="iso:ListIFDs" />
<wsdl:output message="iso:ListIFDsResponse" />
</wsdl:operation>
<!-- GetIFDCapabilities -->
<wsdl:operation name="GetIFDCapabilities">
<wsdl:input message="iso:GetIFDCapabilities" />
<wsdl:output message="iso:GetIFDCapabilitiesResponse" />
</wsdl:operation>
<!-- GetStatus -->
<wsdl:operation name="GetStatus">
<wsdl:input message="iso:GetStatus" />
<wsdl:output message="iso:GetStatusResponse" />
</wsdl:operation>
<!-- Wait -->
<wsdl:operation name="Wait">
<wsdl:input message="iso:Wait" />
<wsdl:output message="iso:WaitResponse" />
</wsdl:operation>
<!-- Cancel -->
<wsdl:operation name="Cancel">
<wsdl:input message="iso:Cancel" />
<wsdl:output message="iso:CancelResponse" />
</wsdl:operation>
<!-- ControlIFD -->
<wsdl:operation name="ControlIFD">
<wsdl:input message="iso:ControlIFD" />
<wsdl:output message="iso:ControlIFDResponse" />
</wsdl:operation>
<!-- ====================== -->
<!-- Card related functions -->
<!-- (portType) -->
<!-- ====================== -->
<!-- Connect -->
<wsdl:operation name="Connect">
<wsdl:input message="iso:Connect" />
<wsdl:output message="iso:ConnectResponse" />
</wsdl:operation>
<!-- Disconnect -->
<wsdl:operation name="Disconnect">
<wsdl:input message="iso:Disconnect" />
<wsdl:output message="iso:DisconnectResponse" />
</wsdl:operation>
<!-- BeginTransaction -->
<wsdl:operation name="BeginTransaction">
<wsdl:input message="iso:BeginTransaction" />
<wsdl:output message="iso:BeginTransactionResponse" />
</wsdl:operation>
<!-- EndTransaction -->
<wsdl:operation name="EndTransaction">
<wsdl:input message="iso:EndTransaction" />
<wsdl:output message="iso:EndTransactionResponse" />
</wsdl:operation>
<!-- Transmit -->
```

```
<wsdl:operation name="Transmit">
<wsdl:input message="iso:Transmit" />
<wsdl:output message="iso:TransmitResponse" />
</wsdl:operation>
<!-- ======================== -->
<!-- User related functions -->
<!-- (portType) -->
<!-- ======================== -->
<!-- VerifyUser -->
<wsdl:operation name="VerifyUser">
<wsdl:input message="iso:VerifyUser" />
<wsdl:output message="iso:VerifyUserResponse" />
</wsdl:operation>
<!-- ModifyVerificationData -->
<wsdl:operation name="ModifyVerificationData">
<wsdl:input message="iso:ModifyVerificationData" />
<wsdl:output message="iso:ModifyVerificationDataResponse" />
</wsdl:operation>
<!-- Output -->
<wsdl:operation name="Output">
<wsdl:input message="iso:Output" />
<wsdl:output message="iso:OutputResponse" />
</wsdl:operation>
</wsdl:portType>
<!-- ======================== -->
<!-- Definition of Binding -->
<!-- ======================== -->
<wsdl:binding name="IFD" type="iso:IFD">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http" />
<!-- ================================ -->
<!-- Card terminal related functions -->
<!-- (binding) -->
<!-- ================================ -->
<!-- EstablishContext -->
<wsdl:operation name="EstablishContext">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:EstablishContext" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ReleaseContext -->
<wsdl:operation name="ReleaseContext">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:ReleaseContext" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ListIFDs -->
<wsdl:operation name="ListIFDs">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:ListIFDs" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
```

```
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- GetIFDCapabilities -->
<wsdl:operation name="GetIFDCapabilities">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:GetIFDCapabilities"
/>
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- GetStatus -->
<wsdl:operation name="GetStatus">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:GetStatus" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- Wait -->
<wsdl:operation name="Wait">
<soap:operation soapAction="urn:iso:std:iso-iec:24727:tech:schema:Wait" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- Cancel -->
<wsdl:operation name="Cancel">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:Cancel" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ControlIFD -->
<wsdl:operation name="ControlIFD">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:ControlIFD" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ====================== -->
<!-- Card related functions -->
<!-- (binding) -->
<!-- ====================== -->
<!-- Connect -->
<wsdl:operation name="Connect">
```

```
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:Connect" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- Disconnect -->
<wsdl:operation name="Disconnect">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:Disconnect" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- BeginTransaction -->
<wsdl:operation name="BeginTransaction">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:BeginTransaction" />
<wsdl:input>
<soap:body use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- EndTransaction -->
<wsdl:operation name="EndTransaction">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:EndTransaction" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- Transmit -->
<wsdl:operation name="Transmit">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:Transmit" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ====================== -->
<!-- User related functions -->
<!-- (binding) -->
<!-- ====================== -->
<!-- VerifyUser -->
<wsdl:operation name="VerifyUser">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:VerifyUser" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
```

```
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- ModifyVerificationData -->
<wsdl:operation name="ModifyVerificationData">
<soap:operation
soapAction="urn:iso:std:isoiec:
24727:tech:schema:ModifyVerificationData" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<!-- Output -->
<wsdl:operation name="Output">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:Output" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<!-- Definition of IFD-Service -->
<wsdl:service name="IFD">
<wsdl:port name="IFDPort" binding="iso:IFD">
<soap:address location="http://127.0.0.1:18080" />
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

پیوست پ

(الزامی)

IFD-Callback-API – اتصال خدمت وب

اتصال خدمت وب برای IFD-Callback-API در زیر، تعریف شده است:

- ISOIFDCallback.XSD – که در آن، پارامترهای درخواست و پاسخ SignalEvent به صورت اجزاء XML، مشخص شده‌اند.

- ISOIFDCallback.WSDL – که دربرگیرنده ISOIFDCallback.XSD است و اتصال خدمت وب برای IFD-Callback-API را مشخص می‌نماید.

پ-۱ مشخصه ISOIFDCallback.XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="urn:iso:std:iso-
iec:24727:tech:schema"
xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
version="1.1">
<!--
<element name="IFDCallbackSchemaVersion">
<complexType attribute name="schemaVersion" type="decimal" use="required"/>
</element>
<!-- Definition of Basic Types -->
<include schemaLocation="ISOIFD.xsd"></include>
<element name="SignalEvent">
<complexType>
<complexContent>
<extension base="iso:RequestType">
<sequence>
<element name="SessionIdentifier" type="string" />
<element name="IFDEvent"
type="iso:IFDStatusType" maxOccurs="unbounded"
minOccurs="0" />
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="SignalEventResponse" type="iso:ResponseType" />
</schema>
```

پ-۲ مشخصه ISOIFDCallback.WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:iso="urn:iso:std:iso-iec:24727:tech:schema"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
version="1.1">
<!--
<!-- ================== -->
<!-- Definition of types -->
<!-- (only include XSDs) -->
<!-- ================== -->
<wsdl:types>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
```

```
targetNamespace="urn:iso:std:iso-iec:24727:tech:schema">
<xsd:include schemaLocation="ISOIFDCallback.xsd" />
</xsd:schema>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
</xsd:schema>
</wsdl:types>
<!-- ====================== -->
<!-- Definition of messages -->
<!-- ====================== -->
<wsdl:message name="SignalEvent">
<wsdl:part name="parameters" element="iso:SignalEvent" />
</wsdl:message>
<wsdl:message name="SignalEventResponse">
<wsdl:part name="parameters" element="iso:SignalEventResponse" />
</wsdl:message>
<!-- ====================== -->
<!-- Definition of portType -->
<!-- ====================== -->
<wsdl:portType name="IFDCallback">
<wsdl:operation name="SignalEvent">
<wsdl:input message="iso:SignalEvent" />
<wsdl:output message="iso:SignalEventResponse" />
</wsdl:operation>
</wsdl:portType>
<!-- ====================== -->
<!-- Definition of Binding -->
<!-- ====================== -->
<wsdl:binding name="IFDCallback" type="iso:IFDCallback">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="SignalEvent">
<soap:operation
soapAction="urn:iso:std:iso-iec:24727:tech:schema:SignalEvent" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<!-- Definition of IFDCallback-Service -->
<wsdl:service name="IFDCallback">
<wsdl:port name="IFDCallbackPort" binding="iso:IFDCallback">
<soap:address location="http://127.0.0.1:18080" />
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

```
ISO24727-4-IFDAPI { iso(1) standard(0) iso24727(24727) part4(4) ifdapi (74) }
-- Version 1.5, 24-May-2010
--
-- IF-PROFILE value '01'
--
-- *According to ISO/IEC 24727-2, the optional IF-PROFILE field in the CCD is
-- used to indicate that a card provides an implementation of ISO/IEC 24727-3.
-- © ISO/IEC 2008-2010
-- All rights reserved. Unless otherwise specified, no part of this publication
-- may be reproduced or utilized in any form or by any means, electronic or
-- mechanical, including photocopying and microfilm, without permission in
-- writing from either ISO at the address below or ISO's member body in the
-- country of the requester.
--
DEFINITIONS AUTOMATIC TAGS EXTENSIBILITY IMPLIED ::=
BEGIN
-- EXPORTS (all)
IMPORTS NonNegativeInt, PositiveInt, GenericHandleType, IFDName, IFDAction,
URIType, IFDSessionIdentifier, TransactionIdentifier
FROM ISO24727-COMMON { iso(1) standard(0) iso24727(24727) };
-- Major and Minor Revision values for this ASN.1 Module
revMajISO24727-4-IFDAPI INTEGER ::= 1
revMinISO24727-4-IFDAPI INTEGER ::= 5
ChannelHandleType ::= SEQUENCE {
protocolTerminationPoint URIType OPTIONAL,
sessionIdentifier GenericIdentifierType OPTIONAL,
binding URIType OPTIONAL
}
IFDNameList ::= SEQUENCE OF IFDName
IFDContextHandle ::= GenericHandleType
IFDSlotHandle ::= GenericHandleType
IFDCallbackChannelHandle ::= GenericHandleType
IFDPadChar ::= NULL
IFDDateTime ::= UTCTime
IFDCapabilities ::= SEQUENCE {
slotCapability SEQUENCE (SIZE (1..1000)) OF IFDSlotCapability,
displayCapability SEQUENCE (SIZE (1..1000)) OF IFDDisplayCapability,
keypadCapability SEQUENCE (SIZE (1..1000)) OF IFDKeypadCapability,
bioSensorCapability SEQUENCE (SIZE (1..1000)) OF IFDBioSensorCapability
opticalSignalUnit BOOLEAN
acousticSignalUnit BOOLEAN
}
IFDInputUnit ::= CHOICE {
pinInput IFDPINInput,
biometricInput IFDBiometricInput
}
IFDPINInput ::= SEQUENCE {
index NonNegativeInt,
passwordAttributes IFDPasswordAttributes
}
IFDPasswordAttributes ::= SEQUENCE {
pwdFlags IFDPasswordFlags,
pwdType IFDPasswordType,
minLength NonNegativeInt,
storedLength NonNegativeInt,
maxLength NonNegativeInt OPTIONAL,
padChar IFDPadChar OPTIONAL,
lastPasswordChange IFDDateTime OPTIONAL
```

```
}
IFDPasswordFlags ::= BIT STRING {
case-sensitive(0),
local(1),
change-disabled(2),
unblock-disabled(3),
initialized(4),
needs-padding(5),
unblockingPassword(6),
soPassword(7),
disable-allowed(8),
integrity-protected(9),
confidentiality-protected(10),
exchangeRefData(11),
resetRetryCounter1(12),
resetRetryCounter2(13)
}
IFDPasswordType ::= CHOICE {
bcd NULL,
ascii-numeric NULL,
utf8 NULL,
half-nibble-bcd
NULL,
iso9564-1 NULL
}
```

```
IFDAltVUMessages ::= SEQUENCE {
authenticationRequestMessage OCTET STRING OPTIONAL,
successMessage OCTET STRING OPTIONAL,
authenticationFailedMessage OCTET STRING OPTIONAL,
requestConfirmationMessage OCTET STRING OPTIONAL,
cancelMessage OCTET STRING OPTIONAL
}
IFDBiometricInput ::= SEQUENCE {
index NonNegativeInt,
biometricSubType NonNegativeInt
}
IFDSlotCapability ::= SEQUENCE {
index NonNegativeInt,
contactBased BOOLEAN
}
IFDSlotStatus ::= SEQUENCE {
index NonNegativeInt,
cardAvailable BOOLEAN,
aTRorATS OCTET STRING OPTIONAL
}
IFDSlotStatusList ::= SEQUENCE OF IFDSlotStatus
IFDDisplayCapability ::= SEQUENCE {
index NonNegativeInt,
lines NonNegativeInt,
columns NonNegativeInt,
virtualLines NonNegativeInt OPTIONAL,
virtualColumns NonNegativeInt OPTIONAL
}
IFDKeypadCapability ::= SEQUENCE {
index NonNegativeInt,
numKeys PositiveInt
}
IFDBioSensorCapability ::= SEQUENCE {
index NonNegativeInt,
biometricType NonNegativeInt
}
IFDStatus ::= SEQUENCE {
iFDName IFDName,
connected BOOLEAN OPTIONAL,
slotStatus IFDSlotStatusList,
activeAntenna BOOLEAN OPTIONAL,
displayStatus IFDSimpleFUStatusList,
keypadStatus IFDSimpleFUStatusList,
bioSensorStatus IFDSimpleFUStatusList
}
IFDStatusList ::= SEQUENCE OF IFDStatus
IFDSimpleFUStatus ::= SEQUENCE {
index NonNegativeInt,
available BOOLEAN
}
IFDSimpleFUStatusList ::= SEQUENCE OF IFDSimpleFUStatus
IFDOutputInfo ::= SEQUENCE {
timeout PositiveInt OPTIONAL,
displayIndex NonNegativeInt OPTIONAL,
message VisibleString OPTIONAL,
acousticSignal BOOLEAN OPTIONAL,
opticalSignal BOOLEAN OPTIONAL
}
-- 7.4.1 IFD_API_EstablishContext
IFDAPIEstablishContextArgument ::= SEQUENCE {
channelHandle ChannelHandleType OPTIONAL
}
IFDAPIEstablishContextResult ::= SEQUENCE {
contextHandle IFDContextHandle OPTIONAL
}
```

```
IFDAPIEstablishContextReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CHANNEL_HANDLE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIEstablishContext ::= SEQUENCE {
arugment IFDAPIEstablishContextArgument,
result IFDAPIEstablishContextResult OPTIONAL,
return IFDAPIEstablishContextReturnCode
}
IFDAPIEstablishContextCall ::= [APPLICATION 4022] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIEstablishContextArgument
}
IFDAPIEstablishContextReturn ::= [APPLICATION 4023] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIEstablishContextResult OPTIONAL,
returnCode IFDAPIEstablishContextReturnCode
}
-- 7.4.2 IFD_API_ReleaseContext
IFDAPIReleaseContextArgument ::= SEQUENCE {
contextHandle IFDContextHandle
}
IFDAPIReleaseContextReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CHANNEL_HANDLE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIReleaseContext ::= SEQUENCE {
arugment IFDAPIReleaseContextArgument,
result NULL OPTIONAL,
return IFDAPIReleaseContextReturnCode
}
IFDAPIReleaseContextCall ::= [APPLICATION 4027] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIReleaseContextArgument
}
IFDAPIReleaseContextReturn ::= [APPLICATION 4028] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIReleaseContextReturnCode
}
-- 7.4.3 IFD_API_ListIFDs
IFDAPIListIFDsArgument ::= SEQUENCE {
contextHandle IFDContextHandle
}
IFDAPIListIFDsResult ::= SEQUENCE {
iFDNameList IFDNameList
}
IFDAPIListIFDsReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIListIFDs ::= SEQUENCE {
arugment IFDAPIListIFDsArgument,
result IFDAPIListIFDsResult OPTIONAL,
return IFDAPIListIFDsReturnCode
}
IFDAPIListIFDsCall ::= [APPLICATION 4032] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIListIFDsArgument
}
```

```
IFDAPIListIFDsReturn ::= [APPLICATION 4033] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIListIFDsResult OPTIONAL,
returnCode IFDAPIListIFDsReturnCode
}
-- 7.4.4 IFD_API_GetIFDCapabilities
IFDAPIGetIFDCapabilitiesArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
iFDName IFDName
}
IFDAPIGetIFDCapabilitiesResult ::= SEQUENCE {
iFDCapabilities IFDCapabilities
}
IFDAPIGetIFDCapabilitiesReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIGetIFDCapabilities ::= SEQUENCE {
arugment IFDAPIGetIFDCapabilitiesArgument,
result IFDAPIGetIFDCapabilitiesResult OPTIONAL,
return IFDAPIGetIFDCapabilitiesReturnCode
}
IFDAPIGetIFDCapabilitiesCall ::= [APPLICATION 4037] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIGetIFDCapabilitiesArgument
}
IFDAPIGetIFDCapabilitiesReturn ::= [APPLICATION 4038] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIListIFDsResult OPTIONAL,
returnCode IFDAPIGetIFDCapabilitiesReturnCode
}
-- 7.4.5 IFD_API_GetStatus
IFDAPIGetStatusArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
iFDName IFDName OPTIONAL
}
IFDAPIGetStatusResult ::= SEQUENCE {
iFDStatus IFDStatus
}
IFDAPIGetStatusReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIGetStatus ::= SEQUENCE {
arugment IFDAPIGetStatusArgument,
result IFDAPIGetStatusResult OPTIONAL,
return IFDAPIGetStatusReturnCode
}
IFDAPIGetStatusCall ::= [APPLICATION 4042] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIGetStatusArgument
}
IFDAPIGetStatusReturn ::= [APPLICATION 4043] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIGetStatusResult OPTIONAL,
returnCode IFDAPIGetStatusReturnCode
}
-- 7.4.6 IFD_API_Wait
IFDAPIWaitArgument ::= SEQUENCE {
```

```
contextHandle IFDContextHandle,
timeOut PositiveInt OPTIONAL,
callbackChannel IFDCallbackChannelHandle OPTIONAL,
iFDStatusList IFDStatusList
}
IFDAPIWaitResult ::= SEQUENCE {
sessionIdentifier VisibleString OPTIONAL,
iFDStatusList IFDStatusList
}
IFDAPIWaitReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIWait ::= SEQUENCE {
arugment IFDAPIWaitArgument,
result IFDAPIWaitResult OPTIONAL,
return IFDAPIWaitReturnCode
}
IFDAPIWaitCall ::= [APPLICATION 4047] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIWaitArgument
}
IFDAPIWaitReturn ::= [APPLICATION 4048] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIWaitResult OPTIONAL,
returnCode IFDAPIWaitReturnCode
}
-- 7.4.7 IFD_API_Cancel
IFDAPICancelArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
sessionIdentifier IFDSessionIdentifier OPTIONAL,
iFDName IFDName
}
IFDAPICancelReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_CANCEL_NOT_POSSIBLE"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPICancel ::= SEQUENCE {
arugment IFDAPICancelArgument,
result NULL OPTIONAL,
return IFDAPICancelReturnCode
}
IFDAPICancelCall ::= [APPLICATION 4052] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPICancelArgument
}
IFDAPICancelReturn ::= [APPLICATION 4053] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPICancelReturnCode
}
-- 7.4.8 IFD_API_ControlIFD
IFDAPIControlIFDArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
iFDName IFDName,
command OCTET STRING
}
IFDAPIControlIFDResult ::= SEQUENCE {
response OCTET STRING
```

```
}
IFDAPIControlIFDReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIControlIFD ::= SEQUENCE {
arugment IFDAPIControlIFDArgument,
result IFDAPIControlIFDResult OPTIONAL,
return IFDAPIControlIFDReturnCode
}
IFDAPIControlIFDCall ::= [APPLICATION 4057] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIControlIFDArgument
}
IFDAPIControlIFDReturn ::= [APPLICATION 4058] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIControlIFDResult OPTIONAL,
returnCode IFDAPIControlIFDReturnCode
}
-- 7.4.9 IFD_API_Connect
IFDAPIConnectArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
iFDName IFDName,
slot NonNegativeInt,
exclusive BOOLEAN OPTIONAL
}
IFDAPIConnectResult ::= SEQUENCE {
slotHandle IFDSlotHandle
}
IFDAPIConnectReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_SLOT"
-- "IFD_SHARING_VIOLATION"
-- "IFD_NO_CARD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIConnect ::= SEQUENCE {
arugment IFDAPIConnectArgument,
result IFDAPIConnectResult OPTIONAL,
return IFDAPIConnectReturnCode
}
IFDAPIConnectCall ::= [APPLICATION 4062] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIConnectArgument
}
IFDAPIConnectReturn ::= [APPLICATION 4063] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIConnectResult OPTIONAL,
returnCode IFDAPIConnectReturnCode
}
-- 7.4.10 IFD_API_Disconnect
IFDAPIDisconnectArgument ::= SEQUENCE {
slotHandle IFDSlotHandle,
action IFDAction OPTIONAL
}
IFDAPIDisconnectReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
```

```
-- "IFD_UNKNOWN_ACTION"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIDisconnect ::= SEQUENCE {
arugment IFDAPIDisconnectArgument,
result NULL OPTIONAL,
return IFDAPIDisconnectReturnCode
}
IFDAPIDisconnectCall ::= [APPLICATION 4067] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIDisconnectArgument
}
IFDAPIDisconnectReturn ::= [APPLICATION 4068] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIDisconnectReturnCode
}
-- 7.4.11 IFD_API_BeginTransaction
IFDAPIBeginTransactionArgument ::= SEQUENCE {
slotHandle IFDSlotHandle
}
IFDAPIBeginTransactionReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIBeginTransaction ::= SEQUENCE {
arugment IFDAPIBeginTransactionArgument,
result NULL OPTIONAL,
return IFDAPIBeginTransactionReturnCode
}
IFDAPIBeginTransactionCall ::= [APPLICATION 4072] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIBeginTransactionArgument
}
IFDAPIBeginTransactionReturn ::= [APPLICATION 4073] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIBeginTransactionReturnCode
}
-- 7.4.12 IFD_API_EndTransaction
IFDAPIEndTransactionArgument ::= SEQUENCE {
slotHandle IFDSlotHandle
}
IFDAPIEndTransactionReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
-- "IFD_NO_TRANSACTION_STARTED"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIEndTransaction ::= SEQUENCE {
arugment IFDAPIEndTransactionArgument,
result NULL OPTIONAL,
return IFDAPIEndTransactionReturnCode
}
IFDAPIEndTransactionCall ::= [APPLICATION 4077] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIEndTransactionArgument
}
IFDAPIEndTransactionReturn ::= [APPLICATION 4078] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIEndTransactionReturnCode
}
-- 7.4.13 IFD_API_Transmit
IFDAPITransmitArgument ::= SEQUENCE {
```

```
slotHandle IFDSlotHandle,
inputAPDU OCTET STRING
}
IFDAPITransmitResult ::= SEQUENCE {
outputAPDU OCTET STRING
}
IFDAPITransmitReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPITransmit ::= SEQUENCE {
arugment IFDAPITransmitArgument,
result IFDAPITransmitResult OPTIONAL,
return IFDAPITransmitReturnCode
}
IFDAPITransmitCall ::= [APPLICATION 4082] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPITransmitArgument
}
IFDAPITransmitReturn ::= [APPLICATION 4083] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPITransmitResult OPTIONAL,
returnCode IFDAPITransmitReturnCode
}
-- 7.4.14 IFD_API_VerifyUser
IFDAPIVerifyUserArgument ::= SEQUENCE {
slotHandle IFDSlotHandle,
inputUnit IFDInputUnit,
displayIndex NonNegativeInt OPTIONAL,
altVUMessages IFDAltVUMessages OPTIONAL,
timeoutUntilFirstKey PositiveInt OPTIONAL,
timeoutAfterFirstKey PositiveInt OPTIONAL,
template OCTET STRING
}
IFDAPIVerifyUserResult ::= SEQUENCE {
response OCTET STRING
}
IFDAPIVerifyUserReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
-- "IFD_UNKNOWN_INPUT_UNIT"
-- "IFD_CANCELLATION_BY_USER"
-- "IFD_UNKNOWN_ERROR"
-- "IFD_UNKNOWN_PIN_FORMAT"
-- "IFD_UNKNOWN_BIOMETRIC_SUBTYPE"
})
IFDAPIVerifyUser ::= SEQUENCE {
arugment IFDAPIVerifyUserArgument,
result IFDAPIVerifyUserResult OPTIONAL,
return IFDAPIVerifyUserReturnCode
}
IFDAPIVerifyUserCall ::= [APPLICATION 4087] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIVerifyUserArgument
}
IFDAPIVerifyUserReturn ::= [APPLICATION 4088] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result IFDAPIVerifyUserResult OPTIONAL,
returnCode IFDAPIVerifyUserReturnCode
}
-- 7.4.15 IFD_API_ModifyVerificationData
IFDAPIModifyVerificationDataArgument ::= SEQUENCE {
```

```
slotHandle IFDSlotHandle,
inputUnit IFDInputUnit,
displayIndex NonNegativeInt OPTIONAL,
altVUMessages IFDAltVUMessages OPTIONAL,
oldReferenceData OCTET STRING OPTIONAL,
timeoutUntilFirstKey PositiveInt OPTIONAL,
timeoutAfterFirstKey PositiveInt OPTIONAL,
repeatInput BOOLEAN OPTIONAL,
template OCTET STRING
}
```

```
IFDAPIModifyVerificationDataReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_SLOT_HANDLE"
-- "IFD_UNKNOWN_INPUT_UNIT"
-- "IFD_CANCELLATION_BY_USER"
-- "IFD_REPEATED_DATA_MISMATCH"
-- "IFD_UNKNOWN_PIN_FORMAT"
-- "IFD_UNKNOWN_BIOMETRIC_SUBTYPE"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIModifyVerificationData ::= SEQUENCE {
arugment IFDAPIModifyVerificationDataArgument,
result NULL OPTIONAL,
return IFDAPIModifyVerificationDataReturnCode
}
IFDAPIModifyVerificationDataCall ::= [APPLICATION 4092] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIModifyVerificationDataArgument
}
IFDAPIVerifyUserModifyVerificationDataReturn ::= [APPLICATION 4093] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIModifyVerificationDataReturnCode
}
-- 7.4.16 IFD_API_Output
IFDAPIOutputArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
iFDName IFDName,
outputInfo IFDOutputInfo
}
IFDAPIOutput ::= SEQUENCE {
arugment IFDAPIOutputArgument,
result NULL OPTIONAL,
return IFDAPIOutputReturnCode
}
IFDAPIOutputReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_DISPLAY_INDEX"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPIOutputCall ::= [APPLICATION 4097] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPIOutputArgument
}
IFDAPIOutputReturn ::= [APPLICATION 4098] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPIOutputReturnCode
}
-- 7.4.17 IFD_API_SignalEvent
IFDAPISignalEventArgument ::= SEQUENCE {
contextHandle IFDContextHandle,
sessionID IFDSessionIdentifier OPTIONAL,
iFDEvent IFDStatusList
}
```

```
IFDAPISignalEvent ::= SEQUENCE {
arugment IFDAPISignalEventArgument,
result NULL OPTIONAL,
return IFDAPISignalEventReturnCode
}
IFDAPISignalEventReturnCode ::= VisibleString (CONSTRAINED BY {
-- "IFD_OK"
-- "IFD_TIMEOUT_ERROR"
-- "IFD_INVALID_CONTEXT_HANDLE"
-- "IFD_UNKNOWN_IFD"
-- "IFD_UNKNOWN_ERROR"
})
IFDAPISignalEventCall ::= [APPLICATION 4102] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument IFDAPISignalEventArgument
}
IFDAPISignalEventReturn ::= [APPLICATION 4103] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode IFDAPISignalEventReturnCode
}
END
```

پیوست ث

(الزامی)

ماژول TCAPI- استاندارد ملی ایران ۴ـ۱۶۳۸۶

```
ISO24727-4-TCAPI { iso(1) standard(0) iso24727(24727) part4(4) tcapi (73) }
-- Version 1.4, 24-May-2010
--
-- IF-PROFILE value '01'
--
-- *According to ISO/IEC 24727-2, the optional IF-PROFILE field in the CCD is
-- used to indicate that a card provides an implementation of ISO/IEC 24727-3.
-- © ISO/IEC 2008-2010
-- All rights reserved. Unless otherwise specified, no part of this
publication
-- may be reproduced or utilized in any form or by any means, electronic or
-- mechanical, including photocopying and microfilm, without permission in
-- writing from either ISO at the address below or ISO's member body in the
-- country of the requester.
DEFINITIONS AUTOMATIC TAGS EXTENSIBILITY IMPLIED ::=
BEGIN
-- EXPORTS (all)
IMPORTS URIType, IFDSessionIdentifier, TransactionIdentifier,
GenericIdentifierType
FROM ISO24727-COMMON { iso(1) standard(0) iso24727(24727) };
-- Major and Minor Revision values for this ASN.1 Module
revMajISO24727-4-TCAPI INTEGER ::= 1
revMinISO24727-4-TCAPI INTEGER ::= 4
ChannelHandleType ::= SEQUENCE {
protocolTerminationPoint URIType OPTIONAL,
sessionIdentifier GenericIdentifierType OPTIONAL,
binding URIType OPTIONAL
}
-- 7.3.1 TC_API_Open
TCAPIOpenArgument ::= SEQUENCE {
remoteAddress OCTET STRING,
channelParams OCTET STRING
}
TCAPIOpenResult ::= SEQUENCE {
channelHandle ChannelHandleType
}
TCAPIOpenReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_NODE_NOT_REACHABLE"
-- "API_UNKNOWN_ERROR"
})
TCAPIOpen ::= SEQUENCE {
argument TCAPIOpenArgument,
result TCAPIOpenResult OPTIONAL,
return TCAPIOpenReturnCode
}
TCAPIOpenCall ::= [APPLICATION 4003] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPIOpenArgument
}
```

```
TCAPIOpenReturn ::= [APPLICATION 4004] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result TCAPIOpenResult OPTIONAL,
returnCode TCAPIOpenReturnCode
}
-- 7.3.2 TC_API_Close
TCAPICloseArgument ::= SEQUENCE {
channelHandle ChannelHandleType
}
TCAPICloseReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_UNKNOWN_ERROR"
-- "API_UNKNOWN_HANDLE"
})
TCAPIClose ::= SEQUENCE {
arugment TCAPICloseArgument,
result NULL OPTIONAL,
return TCAPICloseReturnCode
}
TCAPICloseCall ::= [APPLICATION 4007] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPICloseArgument
}
TCAPICloseReturn ::= [APPLICATION 4008] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode TCAPICloseReturnCode
}
-- 7.3.3 TC_API_Read
TCAPIReadArgument ::= SEQUENCE {
channelHandle ChannelHandleType
}
TCAPIReadResult ::= SEQUENCE {
message OCTET STRING
}
TCAPIReadReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_UNKNOWN_ERROR"
-- "API_WARNING_BUFFER_LENGTH_EXCEEDED"
-- "API_UNKNOWN_HANDLE"
})
TCAPIRead ::= SEQUENCE {
arugment TCAPIReadArgument,
result TCAPIReadResult OPTIONAL,
return TCAPIReadReturnCode
}
TCAPIReadCall ::= [APPLICATION 4012] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPIReadArgument
}
TCAPIReadReturn ::= [APPLICATION 4013] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result TCAPIReadResult OPTIONAL,
returnCode TCAPIReadReturnCode
}
-- 7.3.4 TC_API_Write
```

```
TCAPIWriteArgument ::= SEQUENCE {
channelHandle ChannelHandleType,
message OCTET STRING
}
TCAPIWriteReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_UNKNOWN_ERROR"
-- "API_UNKNOWN_HANDLE"
})
TCAPIWrite ::= SEQUENCE {
arugment TCAPIWriteArgument,
result NULL OPTIONAL,
return TCAPIWriteReturnCode
}
TCAPIWriteCall ::= [APPLICATION 4017] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPIWriteArgument
}
TCAPIWriteReturn ::= [APPLICATION 4018] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode TCAPIWriteReturnCode
}
-- 7.3.5 TC_API_Reset
TCAPIResetArgument ::= SEQUENCE {
channelHandle ChannelHandleType
}
TCAPIResetReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_UNKNOWN_ERROR"
-- "API_UNKNOWN_HANDLE"
})
TCAPIReset ::= SEQUENCE {
arugment TCAPIResetArgument,
result NULL OPTIONAL,
return TCAPIResetReturnCode
}
TCAPIResetCall ::= [APPLICATION 4012] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPIResetArgument
}
TCAPIResetReturn ::= [APPLICATION 4013] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
returnCode TCAPIResetReturnCode
}
-- 7.3.6 TC_API_GetStatus
TCAPIGetStatusArgument ::= SEQUENCE {
channelHandle ChannelHandleType
}
TCAPIGetStatusResult ::= SEQUENCE {
statusString OCTET STRING
}
TCAPIGetStatusReturnCode ::= VisibleString (CONSTRAINED BY {
-- "API_OK"
-- "API_TIMEOUT_ERROR"
-- "API_UNKNOWN_ERROR"
```

```
-- "API_UNKNOWN_HANDLE"
})
TCAPIGetStatus ::= SEQUENCE {
arugment TCAPIGetStatusArgument,
result TCAPIGetStatusResult OPTIONAL,
return TCAPIGetStatusReturnCode
}
TCAPIGetStatusCall ::= [APPLICATION 4017] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
argument TCAPIGetStatusArgument
}
TCAPIGetStatusReturn ::= [APPLICATION 4018] SEQUENCE {
transactionIdentifier TransactionIdentifier OPTIONAL,
result TCAPIGetStatusResult OPTIONAL,
returnCode TCAPIGetStatusReturnCode
}
END
```

پیوست ج

(اطلاعاتی)

کتابنامه

[۱] استاندارد ملی ایران ۱ ـ ۱۶۲۷۴، فناوری اطلاعات ـ اتصال متقابل سامانه‌های باز ـ مدل مرجع پایه ـ مدل پایه

[۲] استاندارد ملی ایران شماره ۱ـ ۸۲۳۱ ، کارت‌های شناسایی ـ شناسایی صادرکنندگان کارت‌ها ـ قسمت اول: سیستم شماره‌گذاری

[۳] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۳-۷۸۱۶، کارت‌های شناسایی - کارت‌های مدار یکپارچه قسمت ۳- کارت‌های دارای اتصالات - واسط الکتریکی و پروتکل‌های انتقال

[۴] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۲ ـ ۸۸۲۴، فناوری اطلاعات -نشانه گذاری قاعده ی نحوی انتزاعی یک -(ASN.1) ویژگی شئ اطلاعاتی

[۵] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ ـ ۸۸۲۵، فناوری اطلاعات ـ قواعد کدبندی نشانه‌گذاری قاعده‌ی نحوی انتزاعی یک (ASN.1) ویژگی قواعد کدبندی پایه (BER) قواعد کدبندی متعارف (CER) و قواعد کدبندی متمایز(DER)

[۶] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۴ ـ ۸۸۲۵، فناوری اطلاعات ـ قواعد کدبندی نشانه‌گذاری قاعده‌ی نحوی انتزاعی یک (ASN.1) قواعد کدبندی XML(XER)

[۷] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۳ـ ۹۷۹۶، فن‌آوری اطلاعات ـ فنون امنیتی - طرح‌های امضای دیجیتال با قابلیت بازیابی پیام ـ قسمت سوم ـ سازوکارهای مبتنی بر لگاریتم گسسته

[۸] استاندارد ملی ایران شماره: ۲ ـ ۱۶۱۹۶، فناوری اطلاعات ـ فنون امنیتی ـ طرح‌های امضای رقمی(دیجیتال) با بازیابی پیام - قسمت ۲: سازوکارهای مبتنی بر تجزیه اعداد صحیح

[۹] استاندارد ملی ایران ـ ایزو ـ آی ای سی ۱ ـ ۹۷۹۷، فناوری اطلاعات -فنون امنیتی -کدهای احراز هویت پیام(MAC) قسمت ۱- سازوکارهای استفاده از رمزگذاری بلوکی

[۱۰] استاندارد ملی ایران شماره ۱ ـ ۱۰۸۲۵، فناوری اطلاعات ـ فنون امنیتی ـ احراز هویت هستار قسمت ۱ـ کلیات

[۱۱] استاندارد ملی ایران شماره۲ ـ ۱۰۸۲۵، فناوری اطلاعات ـ فنون امنیتی -احراز هویت هستارقسمت ۲ـ سازوکارهای استفاده کننده از الگوریتم‌های پوشیده سازی متقارن

[۱۲] استاندارد ملی ایران شماره ۳ ـ ۱۰۸۲۵، فناوری اطلاعات -فنون امنیتی - احراز هویت هستارقسمت ۲ـ سازوکارهای استفاده کننده از الگوریتم‌های پوشیده‌سازی متقارن

[۱۳] استاندارد ملی ایران شماره ۴ ـ ۱۰۸۲۵، فن آوری اطلاعات ـ فنون امنیتی تشخیص هویت نهاد ـ قسمت چهارم- مکانیزم‌های استفاده کننده از یک تابع مقابله رمزنگاری

[۱۴] استاندارد ملی ایران شماره ۵ ـ ۱۰۸۲۵، فناوری اطلاعات ـ فنون امنیتی ـ احراز هویت هستار ـ قسمت ۵: سازوکارهای استفاده‌کننده از فنون دانش_صفر
[۱۵] استاندارد ملی ایران شماره ۶ ـ ۱۰۸۲۵، فناوری اطلاعات ـ فنون امنیتی ـ احراز هویت هستار قسمت ۶ـ سازوکارهای استفاده از انتقال دستی داده‌ها
[۱۶] استاندارد ملی ایران شماره ۹۶۰۰، فن‌آوری اطلاعات –روش‌های امنیتی –حالت‌های عملیاتی یک الگوریتم رمزنگاری قطعه‌ای N بیتی
[۱۷] استاندارد ملی ایران شماره۱ـ ۹۵۹۸، فن‌آوری اطلاعات - روش‌های امنیتی ـ توابع در هم ساز قسمت اول ـ کلیات
[۱۸] استاندارد ملی ایران شماره ۳ ـ ۹۵۹۸، فناوری اطلاعات ـ فنون امنیتی ـ توابع درهم‌ساز ـ قسمت ۳ـ توابع درهم‌ساز اختصاصی
[۱۹] استاندارد ملی ایران شماره ۴ـ ۹۵۹۸، فن‌آوری اطلاعات - روش‌های امنیتی ـ توابع در هم‌ساز قسمت چهارم ـ توابع درهم‌ساز با استفاده از محاسبات پیمانه‌ای
[۲۰] استاندارد ملی ایران ایزو ـ آی ای سی ۳ ـ ۱۰۵۳۶،کارت‌های شناسایی- کارت های مدار(های)یکپارچه بدون تماس(CICCS)ـ قسمت۳- سیگنال‌های الکترونیکی و رویه‌های باز نشاندن
[۲۱] استاندارد ملی ایران شماره ۱ ـ ۱۱۶۸۴، کارت‌های شناسایی - کارت‌های مدار(های) مجتمع غیر تماسی -کارهای جفت‌شدگی قوی ـ قسمت ۱ـ خصوصیات فیزیکی
[۲۲] استاندارد ملی ایران شماره ۳ ـ ۱۰۸۲۲، فناوری اطلاعات ـ فنون امنیتی ـ مدیریت کلید ـ قسمت ۳ـ ساز و کارهای مبتنی بر فنون نامتقارن
[۲۳] استاندارد ملی ایران شماره ۴ ـ۱۰۸۲۲، فن‌آوری اطلاعات ـ فنون امنیتی ـ مدیریت کلیدـ قسمت چهارم ـ مکانیزم مبتنی بر رازهای ضعیف
[۲۴] استاندارد ملی ایران شماره ۲ ـ۱۶۲۹۰، کارت‌های شناسایی - کارت‌های مدار مجتمع بدون تماس - کارت های مجاورتی ـ قسمت ۲ـ توان بسامد رادیویی و واسط سیگنال
[۲۵] استاندارد ملی ایران شماره ۴ ـ۱۶۲۹۰، کارت‌های شناسایی -کارت‌های مدارمجتمع بدون تماس - کارت های مجاورتی ـ قسمت ۴ـ پروتکل انتقال
[۲۶] استاندارد ملی ایران شماره ۱ـ ۱۱۴۹۴، فناوری اطلاعات ـ فنون امنیت امضاهای دیجیتال با پیوست قسمت ۱:کلیات
[۲۷] استاندارد ملی ایران ایزو ـ آی ای سی شماره ۲ ـ ۱۴۸۸۸، فناوری اطلاعات -فنون امنیتی -امضاهای رقمی(دیجیتالی)با پیوست قسمت ۲- سازوکارهای بر پایه عامل‌بندی صحیح
[۲۸] استاندارد ملی ایران ایزو ـ آی ای سی شماره ۳ـ ۱۴۸۸۸، فناوری اطلاعات ـ فنون امنیتی ـ امضاهای رقمی (دیجیتال) با پیوست قسمت ۳ـ سازوکارهای بر پایه لگاریتم گسسته

[۲۹] استاندارد ملی ایران شماره ۱ـ۱۱۶۸۶، کارت‌های شناسایی ـ کارت‌های مدار(های) مجتمع غیر تماسی - کارت‌های مجاورتی (دوربرد) قسمت ۱ـ خصوصیات فیزیکی
[۳۰] استاندارد ملی ایران ۱ـ ۱۰۸۲۴، فن‌آوری اطلاعات ـ فنون امنیتی الگوریتم‌های رمز نگاری ـ قسمت اول ـ کلیات
[۳۱] استاندارد ملی ایران ۳ـ۱۰۸۲۴، فناوری اطلاعات ـ فنون امنیتی ـ الگوریتم‌های رمز نگاری ـ قسمت ۳ ـ رمزهای بلوکی
[۳۲] استاندارد ملی ایران ۴ـ ۱۰۸۲۴، فن‌آوری اطلاعات - فنون امنیتی الگوریتم‌های رمزنگاری ـ قسمت چهارم- رمزگذاری جریانی
[33] ISO 3166-1, Codes for the representation of names of countries and their subdivisions — Part 1: Country codes
[34] ISO/IEC 7816-8:2004, Identification cards — Integrated circuit cards — Part 8: Commands for security operations
[35] ISO/IEC 7816-9:2004, Identification cards — Integrated circuit cards — Part 9: Commands for card management
[36] ISO/IEC 7816-15, Identification cards — Integrated circuit cards — Part 15: Cryptographic information application
[37] ISO/IEC TR 9577:1999, Information technology — Protocol identification in the network layer
[38] ISO/IEC 9945 (all parts), Information technology — Portable Operating System Interface (POSIX)
[39] ISO 9992-2:1998, Financial transaction cards — Messages between the integrated circuit card and the card accepting device — Part 2: Functions, messages (commands and responses), data elements and structures
[40] ISO/IEC 13673:2000, Information technology — Document processing and related communication — Conformance testing for Standard Generalized Markup Language (SGML) systems
[41] ISO/IEC 16262:2002, Information technology — ECMAScript language specification
[42] ISO/IEC 19784-1:2006, Information technology — Biometric application programming interface — Part 1: BioAPI specification
[43] IETF RFC 1738:1994, Uniform Resource Locators (URL)
[44] IETF RFC 1778:1995, The String Representation of Standard Attribute Syntaxes
[45] IETF RFC 2396:1998, Uniform Resource Identifiers (URI):
Generic Syntax
[46] ISO/IEC 9797-2:2011, Information technology -- Security techniques -- Message Authentication Codes (MACs) -- Part 2: Mechanisms using a dedicated hash-function
[47] ISO/IEC 9797-3:2011, Information technology -- Security techniques -- Message Authentication Codes (MACs) -- Part 3: Mechanisms using a universal hash-function
[48] ISO 10118-2: 2010, Information technology -- Security techniques -- Hash-functions -- Part 2 : Hash-functions using an n-bit block cipher

[49] ISO/IEC 10536-2:1995 , Identification cards -- Contactless integrated circuit(s) cards -- Part 2: Dimensions and location of coupling areas
[50] ISO/IEC 11770-1:2010 , Information technology -- Security techniques -- Key management -- Part 1: Framework
[51] ISO/IEC 11770-2:2008 , Information technology -- Security techniques -- Key management -- Part 2: Mechanisms using symmetric techniques
[52] ISO/IEC 11770-5:2011 , Information technology -- Security techniques -- Key management -- Part 5: Group key management
[53] ISO/IEC 14443-1:2008 , Identification cards -- Contactless integrated circuit cards -- Proximity cards -- Part 1: Physical characteristics
[54] ISO/IEC 14443-3:2011, Identification cards -- Contactless integrated circuit cards -- Proximity cards -- Part 3: Initialization and anticollision
[55]ISO/IEC 15693-2:2006 , Identification cards -- Contactless integrated circuit cards -- Vicinity cards -- Part 2: Air interface and initialization
[56] ISO/IEC 15693-3:2009, Identification cards -- Contactless integrated circuit cards -- Vicinity cards -- Part 3: Anticollision and transmission protocol
[57] ISO/IEC 18033-2:2006 , Information technology -- Security techniques -- Encryption algorithms -- Part 2: Asymmetric ciphers