On the design and implementation of the Open eCard App

Detlef Hühnlein[1] · Dirk Petrautzki[2] · Johannes Schmölz[1] · Tobias Wich[1]
Moritz Horsch[1,3] · Thomas Wieland[2] · Jan Eichholz[4] · Alexander Wiesmaier[5]
Johannes Braun[3] · Florian Feldmann[6] · Simon Potzernheim[2] · Jörg Schwenk[6]
Christian Kahlo[7] · Andreas Kühne[8] · Heiko Veit[8]

[1] ecsec GmbH, Sudetenstraße 16, 96247 Michelau
[2] Hochschule Coburg, Friedrich-Streib-Straße 2, 96450 Coburg
[3] Technische Universität Darmstadt, Hochschulstraße 10, 64289 Darmstadt
[4] Giesecke & Devrient GmbH, Prinzregentenstraße 159, 81677 München
[5] AGT Group (R&D) GmbH, Hilpertstraße 20a, 64295 Darmstadt
[6] Ruhr Universität Bochum, Universitätsstraße 150, 44801 Bochum
[7] AGETO Innovation GmbH, Winzerlaer Straße 2, 07745 Jena
[8] Trustable Ltd., Kirchröder Str. 70e, 30625 Hannover

Abstract: The paper at hand discusses the design and implementation of the "Open eCard App", which is a lightweight and open eID client, which integrates major international standards. It supports strong authentication and electronic signatures with numerous common electronic identity cards in desktop as well as mobile environments. The Open eCard App is designed to be as lightweight, usable and modular as possible to support a variety of popular platforms including Android for example. It will be distributed under a suitable open source license and hence may provide an interesting alternative to existing eID clients.

1. Introduction

Against the background of various electronic identity (eID) card projects around the globe (e.g. in the USA [NIST-PIV], Australia [AGIMO-NSF] and Europe [CEN15480]) there have been numerous initiatives in the area of research, development and standardization of eID cards, smart card middleware components and related services.

The German government, for example, has been issuing a new electronic identity card ("neuer Personalausweis", nPA) since November 2010, which features modern cryptographic privacy enhanced protocols [BSI-TR3110] and contactless smart card technology [ISO14443]. In order to support the broad adoption of this innovative security technology the German government initiated a 30 million euro subsidy program in which security bundles ("IT-Sicherheitskits") are provided for German citizen. These security bundles comprise a free eID client [AusweisApp] for computers and a free or discounted smart card terminal. This eID client was announced to support all major platforms, interface devices, and smart cards as specified in the "eCard-API-Framework" [BSI-TR03112], which in turn is based on major international standards such as [ISO24727], [CEN15480] and [OASIS-DSS]. Thus, there have been high expectations with respect to the real world impact of these developments.

However, despite the tremendous political, technical and financial efforts of the German government the practical situation with respect to the secure, easy and ubiquitous use of the different smart cards involved in the German eCard-strategy (cf. [Kowa07] and [HoSt11]) is not yet satisfying. The currently available eID client [AusweisApp], for example, only supports authentication with the German ID card on selected PC-based platforms. Important features such as the support for electronic signature techniques, other smart cards, the Mac OS platform or mobile devices are still lacking. As of today it is not clear whether[1] and when those features will be supported.

In order to solve this problem the authors of the paper at hand have started to design and implement a lightweight alternative, the "Open eCard App", and will publish its source under a suitable open source license. The present contribution discusses selected aspects related to the design and development of this lightweight eID client.

The remainder of the paper is structured as follows: Section 2 provides an overview of related work. Section 3 summarizes the main functional and non-functional requirements for the Open eCard App. Section 4 presents the high level design that has been developed based on the given requirements. Section 5 highlights selected aspects of the module design and related implementation issues. Section 6 closes the paper with an outlook on the next steps and future development.

2. Related Work

In addition to the development of the eCard-API-Framework [BSI-TR03112], the respective standards (e.g. [CEN15480], [ISO24727] and [OASIS-DSS]) and the eID client [AusweisApp] supported by the German government, there have been various related developments which need to be considered here.

First, there are a few alternative proprietary eID clients (cf. [Ageto-AA] and [bos-Autent]), which support the German ID card, or implement a subset of the eCard-API-Framework (cf. [T7-eCard-QES]).

Furthermore, there have been first academic contributions to enable the use of the German ID card on mobile and PC-based platforms.

[EiHü08] discusses the use of [ISO24727] in a mobile environment. In [Hors09], [Kief10], [WHB+11] and [Hors11] an NFC-enabled Java Micro Edition (Java ME) mobile phone is used for mobile authentication with the German ID card. In the [Androsmex] project an Android based NFC-enabled smartphone is used to implement the Password Authenticated Connection Establishment (PACE) protocol (cf. [BSI-TR3110] and [ICAO-PACE]). In [Petr11] a mobile authentication using the German ID card is realized on an Android-based Smartphone using a separate mobile smart card reader. At the University Koblenz-Landau the rosecat project [Jahn11] aims at providing an open source eID client and the [OpenPACE] project at the Humboldt University Berlin aims at providing a cryptographic library which provides support for PACE and parts of the Extended Access Control (EAC) version 2.0 protocol [BSI-TR3110]. While both latter projects have been developed with PCs as primary target in mind, there have

[1] As the size of the different versions of [AusweisApp] ranges between 56.3 MB and 97.6 MB, there are serious doubts whether this eID client may ever be available for mobile devices.

been related contributions, which focus on mobile eID scenarios (cf. [Beil10], [Oepe10], [MMO11] and [MOMR11]).

In addition to the projects in Germany, there have been some open eID related developments in other countries, which are to be considered. The JMRTD project [JMRTD] provides an open source Java implementation of the Machine Readable Travel Documents (MRTD) standards developed by the International Civil Aviation Organization. For the Belgian eID card there is already a Java library [eidlib], a Java applet [eID-Applet] and software for the creation of XML-based signatures [eID-DSS]. [VeLa+09] discusses some security and privacy improvements for the Belgian eID technology. With [MOCCA] an open source environment for the Austrian Citizen Card is available. Recently, an open source middleware for the Portuguese Citizen Card was introduced [MEDI11].

For the generation and verification of XML Advanced Electronic Signatures (XAdES) the [OpenXAdES] project provided a C-based library and in [Gonç10] a Java-based implementation has been developed. The Sirius project [Sirius-SS] develops an open source signing server, which supports the interfaces standardized in [OASIS-DSS].

With respect to smart cards, there are already two open source projects [OpenSC] and [OpenSCDP], which aim at providing a platform-independent framework for the use of smart cards. While [OpenSC] in particular supports cards with Cryptographic Information Application data according to part 15 of [ISO7816], the [OpenSCDP] project provides scripting based support for the German ID card and the German electronic health card for example. For the Android-platform there is an open source secure element evaluation kit [SEEK4Android], which implements [OpenMobile].

There are various frameworks and components in the area of the Security Assertion Markup Language (SAML). [OpenSAML] is a platform-independent and open source representative of them. The use of eID cards in a SAML-environment has been discussed in [EHS09] and [EHMS10]. Corresponding SAML-profiles have been defined in [BSI-TR03130] and [STORK]. The channel binding presented in Section 3.3.10 of Part 7 of [BSI-TR03112] may be used to prevent man-in-the-middle attacks. Unfortunately, this approach is only applicable to cards which feature the Extended Access Control protocol specified in [BSI-TR3110], such as the German ID card for example. In order to provide a secure SAML-binding, which may be used with arbitrary eID cards, the alternatives discussed in [SAML-HoK], [GaLiSc08] and [KSJG10] as well as the TLS-channel binding specified in [RFC5929] may serve as starting points.

For PC-based platforms a trusted computing environment may be realized utilizing a Trusted Platform Module (TPM) and a Trusted Software Stack (TSS). The [jTSS] project provides an open source TSS implementation for the Java Platform. Because the Open eCard App is required to run also on mobile platforms, particularly Android, it is necessary to consider the specific security aspects for this platform in more detail. Android specific attacks have for example been shown in [DaDm+10] and there exist several projects and publications that discuss possible ways to improve the security of Android smartphones (cf. [BDDH+11], [NKZS10], [YZJF11] and [CoNC10]). To provide a robust and trustworthy implementation of the mobile eID client it is also required to consider unconventional attack vectors such as discussed in [AGMB+10] and

[WaSt10]. On the other side there will be mobile device platforms, which are offering enhanced security features like the Trusted Execution Environment (TEE) specified by Global Platform [GP-TEE]. The TEE realizes a secure operating system next to the standard mobile operating system (e.g. Android, iOS, Windows Phone) and hence, can be utilized to secure the mobile eID client. It offers the possibility to install new trusted applications in the field, which are completely separated from each other and applications running outside the trusted execution environment. Trusted applications can securely store data, access secure elements, perform cryptographic operations and protocols and perform secure input and output using the display and keypad of the mobile device.

3. Requirements for the Open eCard App

This section contains the main functional and non-functional requirements of the lightweight Open eCard App, where the key words MAY, SHOULD, SHALL and MUST are used as defined in [RFC2119].

R1. Support for all popular platforms

The Open eCard App MUST be designed to support the most popular client platforms. In addition to PCs based on Windows, Linux or Mac OS this in particular includes NFC-enabled mobile devices, which are for example based on [Android]. On the other side – unlike the clients considered in [EiHü08], [Hors09] and [Hors11] – we do not restrict ourselves to the limited feature set provided by the Java ME platform, but only require that it SHOULD be possible to port our client to such a platform if it turns out to be necessary.

R2. Modularity, open interfaces and extensibility

In order to facilitate the distributed development and portability to different platforms, the Open eCard App MUST consist of suitable modules, which are connected through open interfaces. Those modules SHOULD be designed to minimize the effort of creating a new client application for a specific purpose[2]. For features, which are expected to change over time, such as cryptographic and communication protocols, the Open eCard App SHALL provide suitable extension mechanisms. The basic cryptographic mechanisms SHOULD be provided in form of a standardized cryptographic module to ensure implementation independence and interoperability for cryptographic functions on each supported platform. In particular the Graphical User Interface (GUI), which is expected to be very platform specific, MUST be clearly separated from the other modules.

R3. Architecture based on ISO/IEC 24727

The architecture of the Open eCard App SHALL be based on the international secure element infrastructure standard [ISO24727]. This means in particular that the Interface Device (IFD) API (cf. [ISO24727], part 4) and the Service Access Layer (SAL) API (cf. [ISO24727], part 3) MUST be supported. The IFD Layer SHALL allow to use a wide range of external card terminals, e.g. those based on the PC/SC architecture or

[2] As sketched in [BHWH11] the mobile phone based eID client may serve as smart card terminal for a PC-based eID-client or as standalone system for mobile authentication scenarios.

[OpenMobile], and NFC-modules integrated in mobile phones and SHOULD support TPMs, if present on the platform. The SAL SHALL support arbitrary smart cards, which are described by a CardInfo file according to Section 9 of [CEN15480][3].

R4. Support for electronic signatures and federated identity management

The Open eCard App SHOULD be able to create advanced electronic signatures in standardized formats (cf. [ETSI-101733], [ETSI-101903] and [ETSI-102778]) using the supported eID cards and / or suitable server systems.

R5. Support for federated identity management

The Open eCard App SHOULD support federated identity management protocols according to internationally acknowledged standards, such as [SAML(v2.0)] for example.

R6. Browser integration

The Open eCard App MUST be start- and accessible from within a browser to perform an authentication at web-based services.

R7. Secure components

The Open eCard App MUST utilize the security features of the attached components. This includes the usage of the secure input and output facility of an attached reader as well as the usage of a possibly available secure operating system like the Trusted Execution Environment for mobile devices [GP-TEE].

R8. Security

The Open eCard App MUST be designed in a security aware manner, such that a formal security evaluation, e.g. according to Common Criteria [CC(v3.1)], is possible with modest additional effort. Furthermore the Open eCard App SHALL use the security features provided by attached components. This includes the usage of the secure input and output facility of an attached reader as well as the usage of a possibly available secure operating system like the Trusted Execution Environment [GP-TEE] for mobile devices.

R9. Open source capable

The Open eCard App SHOULD be substantially free of external dependencies. This way it can be released as open source software under a suitable license and there is no regard to take on rights of third parties.

R10. Transparency

The Open eCard App SHOULD provide information to the user about all the existing connections (Internet), access to smart card and other actions.

R11. Stability

The released versions of the Open eCard App SHOULD always be stable, i.e. work without crashes and undesired behaviour.

[3] See http://www.cardinfo.eu for more information about this topic.

R12. High usability and accessible GUI

The design and implementation of a GUI MUST consider platform specific issues to maximize usability and the GUI SHOULD support accessibility features.

4. High Level Design

Based on previous work (e.g. [BSI-TR03112], [Petr11] and [Hors11]) and the requirements above, the high level design depicted in Figure 1 has been developed. It envisages the implementation of the Open eCard App in the Java programming language, making use of the respective architectural concepts. Java is selected mainly because it is supported on all target platforms (R1) and allows applications that can easily be modularized and updated (R2).

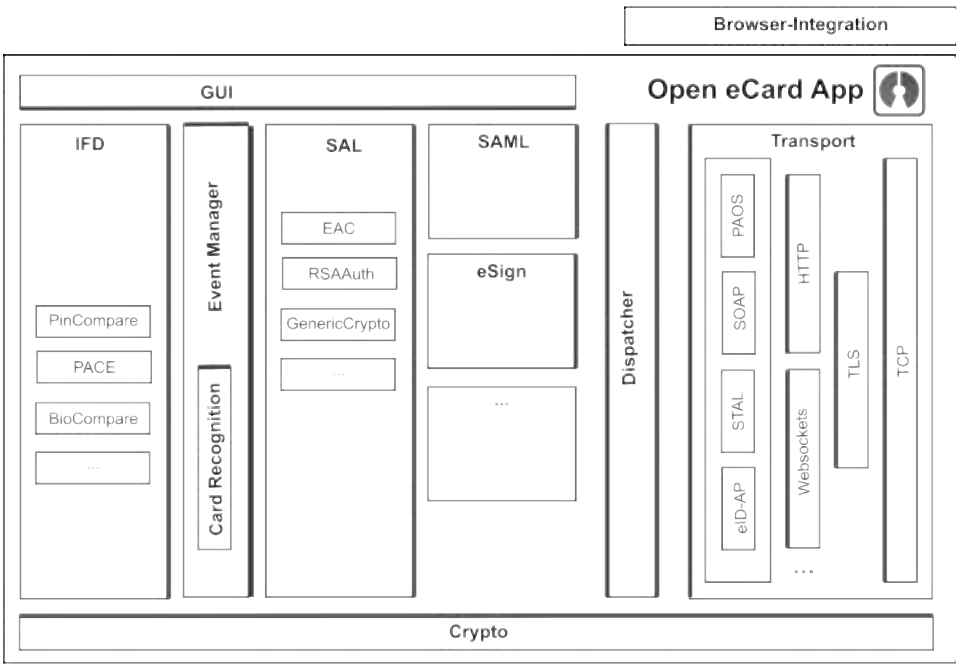

Figure 1: High Level Design of the Open eCard App

The main building blocks of the Open eCard App are as follows:

- Interface Device (IFD)

  This component implements the IFD interface as specified in part 6 of [BSI-TR03112] and part 4 of [ISO24727]. It also contains the additional interfaces for password-based protocols such as PACE (cf. Section 5.1). It provides a generalized interface for communication with specific readers and smart cards, to enable the user to use arbitrary card terminals or smart cards.

- Event Manager

  The Event Manager monitors the occurrence of events (e.g. added or removed terminals or cards) and performs the type-recognition of freshly captured cards (cf. Sections 5.3 - 5.4).

- Service Access Layer (SAL)

  This module implements the Service Access Layer as specified in part 4 of [BSI-TR03112] and part 3 of [ISO24727]. An important aspect of this component is that it provides an extension mechanism, which enables the addition of new authentication protocols in the future without changing other parts of the implementation.

- Crypto

  The crypto module unifies the access to cryptographic functions of the other modules. Through the use of the Java Cryptography Architecture (JCA) [JCA] interface and the provider architecture offered by it, it is easy to exchange the Cryptographic Service Provider (CSP) and hence use the most suitable one for each platform. As JCA provider for the presented eID client primarily [BouncyCastle][4], [FlexiProvider] and [IAIK-JCE] come in mind.

- Graphical User Interface (GUI)

  The GUI is connected via an abstract interface (cf. Section 5.2) and hence is easily interchangeable. This allows providing platform-specific GUI-implementations, while leaving the other modules unchanged.

- Security Assertion Markup Language (SAML)

  This component provides support for the SAML Enhanced Client and Proxy (ECP) profile [SAML-ECP], which allows receiving an `AuthnRequest` via the PAOS-binding [PAOS-v2.0] and starting the eID based authentication procedure with a suitable Identity Provider. Compared to the Web Browser Single Sign-On (SSO) profile used in [BSI-TR03130], the support of the ECP-profile leads to a more straightforward authentication procedure that is easier to protect.

- Electronic Signatures (eSign)

  This component allows to create advanced electronic signatures according to [ETSI-101733], [ETSI-101903] and [ETSI-102778] using the interface defined in part 2 of [BSI-TR03112], which is based on [OASIS-DSS].

- Dispatcher

  The dispatcher provides a centralized entry point for the handling of incoming and outgoing messages. By this centralization, the dispatcher helps to reduce the amount of Java code and the complexity of the Open eCard App.

[4] In order to avoid conflicts with the crippled version of Bouncy Castle integrated in Android, it may turn out to be advantageous to use [SpongeCastle] instead.

- Transport

  The transport component encapsulates the individual transport protocols settled at various transport layers. The layered structure makes it easy to use the protocols needed for a specific use case and to add new protocols. In order to support the currently available eID servers, the protocol stack will at least allow exchanging PAOS messages, which are bound to HTTP and require TLS and TCP/IP to be transported. However the protocol stack of the eID client is designed to additionally support other bindings (e.g. SOAP [SOAP-v1.1]) and alternative protocols such as the Austrian Security Token Abstraction Layer (STAL) [MOCCA] or the eID applet protocol used in Belgium [eID-Applet].

- Browser Integration

  As the Open eCard App should start automatically (without requiring further action by the user) on accessing an appropriately prepared website, there has to be a mechanism for the browser to launch the application and pass over (connection-) parameters. For this purpose, the eID activation mechanisms specified in part 7 of [BSI-TR03112] and the cryptographic interfaces supported by popular browsers (e.g. [PKCS#11]) are implemented in this module.

5. Selected Aspects of the Module Design and Implementation

This section highlights selected aspects of the design and implementation of the Open eCard App.

5.1 PACE in IFD Layer

The standardisation of the IFD interface in part 4 of [ISO24727] took place before all details of the PACE protocol (see [BSI-TR3110] and [ICAO-PACE]) and the corresponding card terminals (see [BSI-TR03119] and [PC/SC-PACE]) were specified. Thus PACE-support is currently not yet integrated in the standardized IFD layer and existing eID clients such as [AusweisApp], [bos-Autent] and [Ageto-AA] seem to implement PACE on top of the IFD layer. As in this case the Service Access Layer needs to be aware of the detailed capabilities of the connected card terminal this is not optimal from an architectural point of view.

To overcome this problem we propose an extension for the IFD API, that contains the two commands `EstablishChannel` and `DestroyChannel`, which are protocol agnostic generalizations of the `EstablishPACEChannel` and `DestroyPACE-Channel` commands defined in [PC/SC-PACE] and (partly) in [BSI-TR03119].

5.2 GUI Interface

As the Graphical User Interface (GUI) needs to be developed in a platform specific manner, it is necessary to introduce an interface, which decouples the user interface from the rest of the eID client. As the Open eCard App shall support a wide range of smart card terminals with varying capabilities in a homogeneous manner, the GUI needs to compensate these differences and provide a card- and terminal-specific dialogue to obtain the user consent for a specific transaction. In order to support arbitrary terminals and eID cards, the GUI interface is defined in an abstract manner. A user dialogue specification consists of a sequence of steps, which in turn may contain a sequence of input and output elements. The input elements allow to mark check boxes, which may for example be used to restrict a Certificate Holder Authorization Template (cf. [BSI-TR3110], Annex C.1.5 and C.4), or capture a PIN.

5.3 Event Manager

Events in the IFD layer (e.g. insertion or removal of cards) can be recognized using the `Wait` function of the IFD interface as specified in part 6 of [BSI-TR03112] and in part 4 of [ISO24727]. This function returns the state of a monitored terminal, after an event has occurred. In order to identify a specific event, the calling component must compare the received state with the previous state of the terminal. Thus, every component that makes use of the `Wait` function would need to implement this kind of comparison, which is not very convenient.

To centralize this functionality, we introduce an Event Manager, which monitors the occurrence of events, triggers the card recognition and distributes the received information to all components that have registered to it. A component can register for one or more specific events (e.g. insertion of cards) and will be notified if one of them occurs. Furthermore custom filters can be applied to the Event Manager, in case the predefined registration options are not sufficient.

5.4 Card Recognition

In order to support the widest possible range of eID cards, the Open eCard App supports CardInfo structures according to [BSI-TR03112] Part 4, Annex A and [CEN15480] Section 9. For the recognition of the card type it is necessary to construct a decision tree (cf. [BSI-TR03112] Part 4, Figure 5 and [Wich11] Section 5.1.2) using the set of available CardInfo files. While this construction could be performed by an eID client upon initialization, we propose to perform this construction only once and store the constructed decision tree in a suitable XML format. As there is no need for the eID client to perform the construction itself, we propose that a suitable infrastructure component, such as the CardInfo repository (cf. [BSI-TR03112] Part 5), performs the construction and distributes the compact decision tree.

The Card Recognition module within the Open eCard App (cf. Figure 1) works with the recognition tree and just needs access to the IFD. As soon as a new card is captured and a corresponding event is identified by the Event Manager (cf. Section 5.3), the card recognition procedure is performed by connecting the card and walking through the recognition tree until the card type is determined. In the eCard-API layering model, this request to the Card Recognition module is performed by the SAL. However, with the availability of the Event Manager, the most natural approach is to perform the recognition right after a *"card inserted"* event and distribute the information with a *"card recognised"* event afterwards. This information distribution mechanism has the advantage that not only the SAL, but also other modules which need this kind of information (e.g. the GUI), can act as an event sink, too.

6. Summary

The paper at hand presents the design and selected implementation details of the Open eCard App. This eID client supports arbitrary smart cards, which are described by CardInfo files and is designed to support PC-based as well as mobile platforms, e.g. based on [Android]. As the Open eCard App is designed to be as lightweight and usable as possible and will be distributed under a suitable open source license, it may provide an interesting open alternative to the currently available eID clients such as the [AusweisApp].

7. References

[Ageto-AA] Ageto Innovation GmbH: *AGETO AusweisApp*, http://www.ageto.de/egovernment/ageto-ausweis-app

[AGIMO-NSF] Australian Government Information Management Office (AGIMO): *National Smartcard Framework*, http://www.finance.gov.au/e-government/security-and-authentication/smartcard-framework.html

[AGMB+10] A. Aviv, K. Gibson, E. Mossop, M. Blaze and J. Smith: *Smudge attacks on smartphone touch screens*, WOOT'10 Proceedings of the 4th USENIX conference on offensive technologies, http://www.usenix.org/events/woot10/tech/full_papers/Aviv.pdf

[Android] Google Inc.: *Android Website*, http://www.android.com/

[Androsmex] T. Senger & al.: *Androsmex Project - A mobile smart card explorer for android smartphones with NFC capabilities*, http://code.google.com/p/androsmex/

[AusweisApp] BSI: *Official Portal for the eID-client "AusweisApp"*, http://www.ausweisapp.de

[BDDH+11] S. Bugiel, L. Davi, A. Dmitrienko, S. Heuser, A. Sadeghi and B. Shastry: *Practical and Lightweight Domain Isolation on Android,* Proceedings of the 1st ACM CCS Workshop on Security and Privacy in Mobile Devices (SPSM), ACM Press, October 2011, http://www.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_TRUST/PubsPDF/spsm18-bugiel.pdf

[Beil10] K. Beilke: *Mobile eCard-API*, Humboldt-University, Diploma Thesis, 2010, http://sar.informatik.hu-berlin.de/research/publications/#SAR-PR-2010-12

[BHWH11] J. Braun, M. Horsch, A. Wiesmaier and D. Hühnlein: *Mobile Authentication and Signature (in German),* DACH Security 2011, pp. 1-12, http://www.cdc.informatik.tu-darmstadt.de/mona/pubs/201109_DACH11_Mobile_Authentisierung_und_Signatur.pdf

[BouncyCastle] The Legion of the Bouncy Castle: *Bouncy Castle API*, http://www.bouncycastle.org/

[bos-Autent] bos GmbH & Co. KG: *Governikus Autent*, http://www.bos-bremen.de/de/governikus_autent/1854605/

[BSI-TR3110] BSI: *Advanced Security Mechanism for Machine Readable Travel Documents – Extended Access Control (EAC)*, Technical Directive of the Federal Office for Information Security Nr. 03110, BSI TR-03110, Version 2.05, https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03110/index_htm.html

[BSI-TR03112] BSI: *eCard-API-Framework*, Technical Directive of the Federal Office for Information Security Nr. 03112, BSI TR-03112, Version 1.1, https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03112/index_htm.html

[BSI-TR03119] BSI: *Requirements for Card Terminals with support for the new German eID-card*, in German, Technical Directive of the Federal Office for Information Security Nr. 03119, BSI TR-03119, Version 1.2, 27.03.2011, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03119/BSI-TR-03119_V1_pdf

[BSI-TR03130] BSI: *eID-Server*, Technical Directive of the Federal Office for Information Security Nr. 03130 (in German), BSI TR-03130, Version 1.4.1, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03130/TR-03130_TR-eID-Server_V1_4_pdf

[CC(v3.1)] CCMB: *Common Criteria for Information Technology Security Evaluation*, Version 3.1, part 1-3, 2009, http://www.commoncriteriaportal.org/cc/

[CEN15480] CEN: *Identification card systems — European Citizen Card*, CEN TS 15480 (Part 1-4)

[CoNC10] M. Conti1, V. Nguyen and B. Crispo: *CRePE - Context-Related Policy Enforcement for Android*, ISC'10 Proceedings of the 13th international conference on Information security, Springer, ISBN: 978-3-642-18177-1, http://www.few.vu.nl/~mconti/papers/C16.pdf

[DaDm+10] L. Davi, A. Dmitrienko, A. Sadeghi and M. Winandy: *Privilege Escalation Attacks on Android*, ISC'10 Proceedings of the 13th international conference on Information security, Springer, ISBN: 978-3-642-18177-1, http://www.ei.rub.de/media/trust/veroeffentlichungen/2010/11/13/DDSW2010_Privilege_Escalation_Attacks_on_Android.pdf

[eID-Applet] F. Cornelis & al.: *eID-Applet Project*, http://code.google.com/p/eid-applet/

[eID-DSS] F. Cornelis & al.: *eID Digital Signature Service Project*, http://code.google.com/p/eid-dss/

[eidlib] K. Overdulve: *eidlib Project*, http://code.google.com/p/eidlib/

[EHS09] J. Eichholz, D. Hühnlein and J. Schwenk: *SAMLizing the European Citizen Card*, in A. Brömme & al. (Ed.), BIOSIG 2009: Biometrics and Electronic Signatures, GI-Edition Lecture Notes in Informatics (LNI) 155, 2009, pp. 105-117, http://www.ecsec.de/pub/SAMLizing-ECC.pdf

[EHMS10] J. Eichholz, D. Hühnlein, G. Meister and J. Schmölz: *New Authentication concepts for electronic Identity Tokens*, in Proceedings of "ISSE 2010", Vieweg, 2010, pp. 26-38, http://www.ecsec.de/pub/ISSE2010.pdf

[EiHü08] J. Eichholz and D. Hühnlein: *Using ISO/IEC 24727 for mobile devices*, in Proceedings of Sicherheit 2008, GI, LNI 128, pp. 581-587, http://www.ecsec.de/pub/2008_Sicherheit.pdf

[ETSI-101733] ETSI: *CMS Advanced Electronic Signatures (CAdES)*, ETSI TS 101 733, Version 1.8.1. http://pda.etsi.org/pda/queryform.asp, December 2009

[ETSI-101903] ETSI: *Technical Specification XML Advanced Electronic Signatures (XAdES)*, ETSI TS 101 903, Version 1.4.1, http://pda.etsi.org/pda/queryform.asp, June 2009

[ETSI-102778] ETSI: *PDF Advanced Electronic Signature Profiles*, ETSI TS 102 778, part 1-5, http://pda.etsi.org/pda/queryform.asp, 2009

[FlexiProvider] M. Maurer & al.: *Flexiprovider Project*, http://www.flexiprovider.de

[GaLiSc08] S. Gajek, L. Liao and J. Schwenk: *Stronger TLS Bindings for SAML Assertions and SAML Artifacts*, Proceedings of the 2008 ACM workshop on Secure web services

[Gonç10] L. Gonçalves: *XAdES4j— a Java Library for XadES Signature Service*s, Master Thesis, Instituto Superior de Engenharia de Lisboa, 2010, http://luisfsgoncalves.files.wordpress.com/2011/01/xades4j.pdf

[GP-TEE] Global Platform: *The Trusted Execution Environment: Delivering Enhanced Security at a Lower Cost to the Mobile Market*, Whitepaper, February 2011, http://www.globalplatform.org/documents/GlobalPlatform_TEE_White_Paper_Feb2011.pdf

[Hors09] M. Horsch: *MobilePACE – Password Authenticated Connection Establishment implementation on mobile devices*, Bachelor Thesis, TU Darmstadt, 2009, http://www.cdc.informatik.tu-darmstadt.de/mona/pubs/200909_BA_MobilePACE.pdf

[Hors11] M. Horsch: *MONA - Mobile Authentication with the new German eID-card (in German)*, Master Thesis, TU Darmstadt, 2011, http://www.cdc.informatik.tu-darmstadt.de/mona/pubs/201107_MA_Mobile%20Authentisierung%20mit%20dem%20neuen%20Personalausweis%20(MONA).pdf

[HoSt11] M. Horsch and M. Stopczynski: *The German eCard-Strategy*, Technical Report: TI-11/01, TU Darmstadt, http://www.cdc.informatik.tu-darmstadt.de/reports/reports/the_german_ecard-strategy.pdf

[IAIK-JCE] TU Graz: *IAIK Provider for the Java™ Cryptography Extension (IAIK-JCE)*, http://jce.iaik.tugraz.at/

[ICAO-PACE] ICAO: *Supplemental Access Control for Machine Readable Travel Documents*, ICAO Technical Report, Version 1.01, 11.11.2010, http://www2.icao.int/en/MRTD/Downloads/Technical%20Reports/Technical%20Report.pdf

[ISO7816] ISO/IEC: *Identification cards – Integrated circuit cards*, ISO/IEC 7816 (part 1-15)

[ISO14443] ISO/IEC: *Contactless integrated circuit cards - Proximity cards,* ISO/IEC 14443 (Part 1-4)

[ISO24727] ISO/IEC: *Identification cards – Integrated circuit cards programming interfaces*, ISO/IEC 24727 (Part 1-5)

[Jahn11] N. Jahn: *rosecat - Architecture and Implementation of an Open Source eID Client*, in German, Diploma Thesis, University Koblenz-Landau, 2011, http://kola.opus.hbz-nrw.de/volltexte/2011/672/pdf/Diplomarbeit.pdf

[JCA] Oracle: *Java ™ Cryptography Architecture (JCA) Reference Guide*, http://download.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html

[JMRTD] M. Oostdijk & al.: *JMRTD Project*, http://jmrtd.org/

[jTSS] M. Pirkner, R. Toegl & al.: *Trusted Computing for the Java Platform Project*, http://sourceforge.net/projects/trustedjava/

[Kief10] F. Kiefer: *Efficient Implementation of the PACE and EAC Protocol for mobile devices,* in German, Bachelor Thesis, TU Darmstadt, 2010, http://www.cdc.informatik.tu-darmstadt.de/mona/pubs/201007_BA_Effiziente_Implementierung_des_PACE-_und_EAC_Protokolls_fuer_mobile_Geraete.pdf

[Kowa07] B. Kowalski: *The eCard-Strategy of the Federal Government of Germany,* in German, in BIOSIG 2007: Biometrics and Electronic Signatures, Proceedings of the Special Interest Group on Biometrics and Electronic Signatures, LNI 108, pp. 87–96, 2007, http://subs.emis.de/LNI/Proceedings/Proceedings108/gi-proc-108-008.pdf

[KSJG10] F. Kohlar, J. Schwenk, M. Jensen and S. Gajek: *Secure Bindings of SAML Assertions to TLS Sessions*, Proceedings of the Fifth International Conference on Availability, Reliability and Security (ARES), 2010

[MEDI11] L. Medinas: *The development of the new Free/Opensource Portuguese Citizen Card Middleware*, https://codebits.eu/intra/s/proposal/212

[MMO11] W. Müller, F. Morgner and D. Oepen: *Mobile scenario for the new German ID card*, in German, in 21st Smartcard-Workshop, 2.-3. Februar 2011, Darmstadt, pp. 179-188, 2011, http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2011-01/SAR-PR-2011-01.pdf

[MOCCA] MOCCA: *Modular Open Citizen Card Architecture Project*, http://mocca.egovlabs.gv.at/

[MOMR11] F. Morgner, D. Oepen, W. Müller and J.-P. Redlich: *Mobile Reader for the new German ID card*, in German, in 12th German IT-Security Congress, SecuMedia, pp. 227-240, 2011, http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2011-04/SAR-PR-2011-04.pdf

[NIST-PIV] NIST: *About Personal Identity Verification (PIV) of Federal Employees and Contractors*, http://csrc.nist.gov/groups/SNS/piv/index.html

[NKZS10] M. Nauman , S. Khan , X. Zhang and J. Seifert: *Beyond Kernel-level Integrity Measurement: Enabling Remote Attestation for the Android Platform*, In Trust and Trustworthy Computing, Vol. 6101 (2010), http://profsandhu.com/zhang/pub/trust10-android.pdf

[OASIS-DSS] OASIS: *Digital Signature Service Core Protocols, Elements, and Bindings*, Version 1.0, OASIS Standard, via http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-spec-v1.0-os.pdf

[Oepe10] D. Oepen: *Authentication in the mobile web – on the usability of eID based authentication using an NFC based mobile device*, in German, Diploma Thesis, Humboldt-University, 2010, http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2010-11/SAR-PR-2010-11_.pdf

[OpenPACE] F. Morgner & al.: *OpenPACE Project - Crypto library for the PACE protocol,* http://openpace.sourceforge.net/

[OpenMobile] SIM Card Alliance: *Open Mobile API specification,* Version 1.2, http://tinyurl.com/ckl7sbt

[OpenSAML] S. Cantor & al.: *OpenSAML Project*, https://wiki.shibboleth.net/confluence/display/OpenSAML/Home

[OpenSC] M. Paljak & al.: *OpenSC Project - tools and libraries for smart card,* http://www.opensc-project.org

[OpenSCDP] A. Schwier & al.: *OpenSCDP Project - Open Smart Card Development Platform,* http://www.openscdp.org/

[OpenXAdES] T. Martens & al.: *OpenXAdES Project*, http://www.openxades.org/

[PAOS-v2.0] Liberty Alliance Project: *Liberty Reverse HTTP Binding for SOAP Specification*, Version v2.0, via http://www.projectliberty.org/liberty/content/download/909/6303/file/liberty-paos-v2.0.pdf

[PC/SC-PACE] PC/SC Workgroup: *Interoperability Specification for ICCs and Personal Computer Systems - Part 10 IFDs with Secure PIN Entry Capabilities – Amendment 1*, 2011, http://www.pcscworkgroup.com/specifications/files/pcsc10_v2.02.08_AMD1.pdf

[Petr11] D. Petrautzki: *Security of Authentication Procedures for Mobile Devices*, (in German), Master Thesis, Hochschule Coburg, 2011

[PKCS#11] RSA Laboratories: *PKCS #11 Base Functionality v2.30: Cryptoki – Draft 4*, 10 July 2009

[RFC2119] S. Bradner: *Key words for use in RFCs to Indicate Requirement Levels*, IETF RFC 2119, via http://www.ietf.org/rfc/rfc2119.txt

[RFC5929] J. Altman, N. Williams, L. Zhu: *Channel Bindings for TLS*, IETF RFC 5929, via http://www.ietf.org/rfc/rfc5929.txt

[SAML(v2.0)] S. Cantor, J. Kemp, R. Philpott and E. Maler: *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0*, OASIS Standard, 15.03.2005. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf, 2005

[SAML-HoK] N. Klingenstein: *SAML V2.0 Holder-of-Key Web Browser SSO Profile*, OASIS Committee Draft 02, 05.07.2009. http://www.oasis-open.org/committees/download.php/33239/sstc-saml-holder-of-key-browser-sso-cd-02.pdf, 2009

[SAML-ECP] S. Cantor & al.: *SAML V2.0 Enhanced Client or Proxy Profile Version 2.0*, Working Draft 02, 19.02.2011, http://www.oasis-open.org/committees/download.php/41209/sstc-saml-ecp-v2.0-wd02.pdf

[SEEK4Android] F. Schäfer & al.: *Secure Element Evaluation Kit for the Android platform Project*, http://code.google.com/p/seek-for-android/

[Sirius-SS] A. Kühne, H. Veit & al.: *Sirius Sign Server Project*, http://sourceforge.net/projects/sirius-sign/

[SOAP-v1.1] W3C Note: *Simple Object Access Protocol (SOAP) 1.1*, 8 May 2000, via http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[SpongeCastle] R. Tyley: *Sponge Castle Project*, https://github.com/rtyley/spongycastle

[STORK] J. Alcalde-Moraño, J. L. Hernández-Ardieta, A. Johnston, D. Martinez, B. Zwattendorfer: *STORK Deliverable D5.8.1b – Interface Specification*, 08.09.2009, https://www.eid-stork.eu/index.php?option=com_processes&Itemid=&act=streamDocument&did=960

[T7-eCard-QES] T7 e.V.: *Common PKI – Signature API,* in German, http://www.t7ev.org/themen/anwendungsanbieter/common-pki-signatur-api.html

[VeLa+09] P. Verhaeghe, J. Lapon, B. De Decker, V. Naessens and K. Verslype: *Security and privacy improvements for the Belgian eID technology*, 2009, Springer, Emerging Challenges for Security, Privacy and Trust vol:297 pages:237-247, 4th IFIP International Information Security Conference (SEC) edition:24 location:Pafos, Cyprus date:18-20 May 2009, ISBN: 978-3-642-01243-3, https://lirias.kuleuven.be/bitstream/123456789/215296/1/paper.pdf

[WaSt10] Z. Wang, A. Stavrou: *Exploiting Smart-Phone USB Connectivity For Fun And Profit*, ACSAC '10, Proceedings of the 26th Annual Computer Security Applications Conference, www.cs.gmu.edu/~astavrou/research/acsac10.pdf

[WHB+11] A. Wiesmaier, M. Horsch, J. Braun, F. Kiefer, D. Hühnlein, F. Strenzke and J. Buchmann: *An efficient PACE Implementation for mobile Devices,* ASIA CCS '11: 6th ACM Symposium on Information, Computer and Communications Security, vol. ACM Symposium on Information, Computer and Communications Security, p. 176-185, ACM, March 2011, https://www.cdc.informatik.tu-darmstadt.de/de/publikations-details/?no_cache=1&pub_id=TUD-CS-2011-0064

[Wich11] T. Wich: *Tools for automated utilisation of Smart-Card Descriptions*, Master Thesis, Hochschule Coburg, 2011

[YZJF11] Y. Zhou, X. Zhang, X. Jiang and V. Freeh: *Taming Information-Stealing Smartphone Applications (on Android)*, 4th International Conference on Trust and Trustworthy Computing, Pittsburgh, http://online.wsj.com/public/resources/documents/TISSA042511.pdf