H. Roßnagel et al. (Eds.): Open Identity Summit 2019,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2019 25

How to harmonise local and remote signing

Detlef Hühnlein[1], Tobias Wich[1], Tina Hühnlein[1], Sebastian Schuberth[1], René Lottes[1], Neil Crossley[1] and Florian Otto[1]

Abstract: While the generation of qualified electronic signatures traditionally required the use of local qualified electronic signature creation devices (QSCD) in form of smart cards for example, the eIDAS-Regulation [EU14] introduced the promising option for Hardware Security Module (HSM) based QSCDs and remote signature protocols, which are especially suitable for mobile environments. As the technical interfaces of these two approaches are fundamentally different, one until today needs to choose a solution, which either supports local or remote signing but not both. In this paper we show how to harmonise the two seemingly distinct worlds in order to enable adaptive signing solutions which seamlessly allow to use both local and remote QSCDs and provide the best possible user experience for the generation of qualified electronic signatures.

Keywords: local signature creation, remote signature creation, ChipGateway protocol, eIDAS, qualified electronic signature creation device (QSCD)

1 Introduction

An important concept of Regulation (EU) No. 910/2014, which is commonly known as eIDAS-Regulation [EU14], is the qualified electronic signature (Article 3 (12)), which by definition is an advanced electronic signature (Article 26) that is created by a qualified electronic signature creation device (QSCD) (Article 3 (23) and Annex II), and is based on a qualified certificate for electronic signatures (Annex I).

In practice there are two major forms of QSCD:

1. "Local QSCD", which support conventional local signature generation and which may be implemented in form of a smart card for example, which has been evaluated according to [EN14], and
2. "Remote QSCD" according to [EN18b] and [EN18c], which comprises a hardware security module according to [EN18a] and which is operated in the secure environment of a qualified trust service provider.

While the two forms of QSCDs share some similarity in a rather abstract and high-level perspective, the detailed technical behaviour and the corresponding interfaces are very different and hence the technical standards for accessing and using the two forms of QSCDs within signing services available today are fundamentally different, as outlined in

[1] ecsec GmbH, Sudetenstraße 16, 96247 Michelau, {detlef.huehnlein, tobias.wich, tina.huehnlein, sebastian.schuberth, rene.lottes, neil.crossley, florian.otto}@ecsec.de

Section 2. Against this background the present paper introduces in Section 3 a novel approach for harmonising local and remote signing based on a rather simple generalisation of the "ChipGateway protocol", which has been developed in a joint effort by ecsec GmbH and LuxTrust S.A. for local signing in web-based environments and which has been contributed in [LE17] to OASIS TC DSS-X for the purpose of standardisation. In Section 4 we show that the proposed standard-based authentication strategy enables smart enrolment processes in line with Art. 24 (1) of the eIDAS-Regulation [EU14]. Section 5 closes this contribution by summarising the main aspects and providing an outlook on possible future developments.

2 Existing interfaces and protocols for local and remote signing

2.1 Local Signing

A rather complete survey of interfaces for local signing, which were existing in 2005 is contained in Section 2.3 of the German paper [HÜ05]. Among the interfaces, which enjoy practical relevance today are [PC13], which provides a low-level interface to connect to smart card terminals and smart cards, [GF15], which offers a high-level interface to access cryptographic modules, and [MS18], which is similar, but not platform independent and tightly integrated into Microsoft platforms.

Against the background of these interfaces and their lack to support more sophisticated eID-cards and related protocols, such as Extended Access Control v2 [TR15a] for example, the eCard-API-Framework [TR15b] and the related international standard [IS14] were developed. While this standard supports distributed authentication protocols, the signing functionality is purely local.

On the other hand, there is an extension [NC15] of [DR07], which allows to use the distributed signing protocol standardised in [DR07] together with local signature creation devices.

The ChipGateway protocol [LE17] may be considered as a variant of [NC15], which additionally has been inspired by [TR17]. This protocol is not only used for creating qualified electronic signatures in web-based environments, but also for electronic authentication and identification with the Luxembourgish eID card, which recently has been successfully peer reviewed on Level of Assurance "high" (see Art. 8 of [EU14] and Art. 10 of [EU15]).

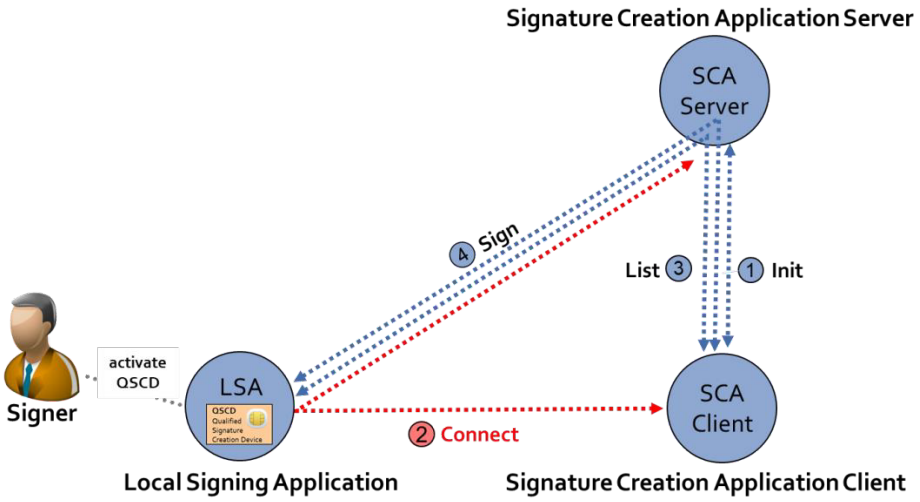

Fig. 1: Outline of ChipGateway protocol

As depicted in Fig. 1, the ChipGateway system consists of

- a "Local Signing Application" (LSA)[2], which is connected to the "Local QSCD" of the Signer and
- a distributed "Signature Creation Application" (SCA), which in turn consists of a "Signature Creation Application Client" (SCA Client) and "Signature Creation Application Server" (SCA Server), which interact using [DR07] or [KH18b].

A typical signing procedure consists of four phases:

1. **`Init`** – In this phase the SCA Client initiates the process by sending an appropriate request[3] to the SCA Server, which returns a `SessionIdentifier` and the ChipGateway endpoint of the SCA Server (`ServerAddress`) in the corresponding `SignResponse`.
2. **`Connect`** – In this phase the SCA Client activates the LSA using a localhost link as in [TR17], which triggers the establishment of a secure connection with the SCA Server, such that it afterwards can send appropriate commands to the LSA. Further details of the connection establishment within the ChipGateway protocol are depicted in Fig. 2 and described below.
3. **`List`** – This phase allows to determine the set of local signature creation devices, which are connected to the LSA using `ListTokensRequest`, which yields a sequence of `TokenInfo` structures, as well as the available certificates (`CertificateInfo`) for signature generation using

[2] The term "Local Signing Application" (LSA) is derived from the term „Server Signing Application" (SSA), used in [EN18b] and [EN18c]. This component was called "ChipGateway" in the [LE17] contribution.
[3] Based on [DR07] or [KH18b], this could be an appropriately profiled `SignRequest`.

`ListCertificatesRequest`. At the end of this phase, the Signer is able to select the private key and certificate, which is to be used for generating the signature in the next phase.

4. **`Sign`** – If this has not happened before, the SCA Client sends the document, which is to be signed to the SCA Server in a `SignRequest`. The SCA Server calculates the hash of the appropriately prepared document and sends a `SignRequest` to the LSA, which finally uses the Local QSCD to create the digital signature. This raw digital signature is returned to the SCA Server, which finally extends it towards an advanced electronic signature according to {C,X,P}AdES[4], if required.

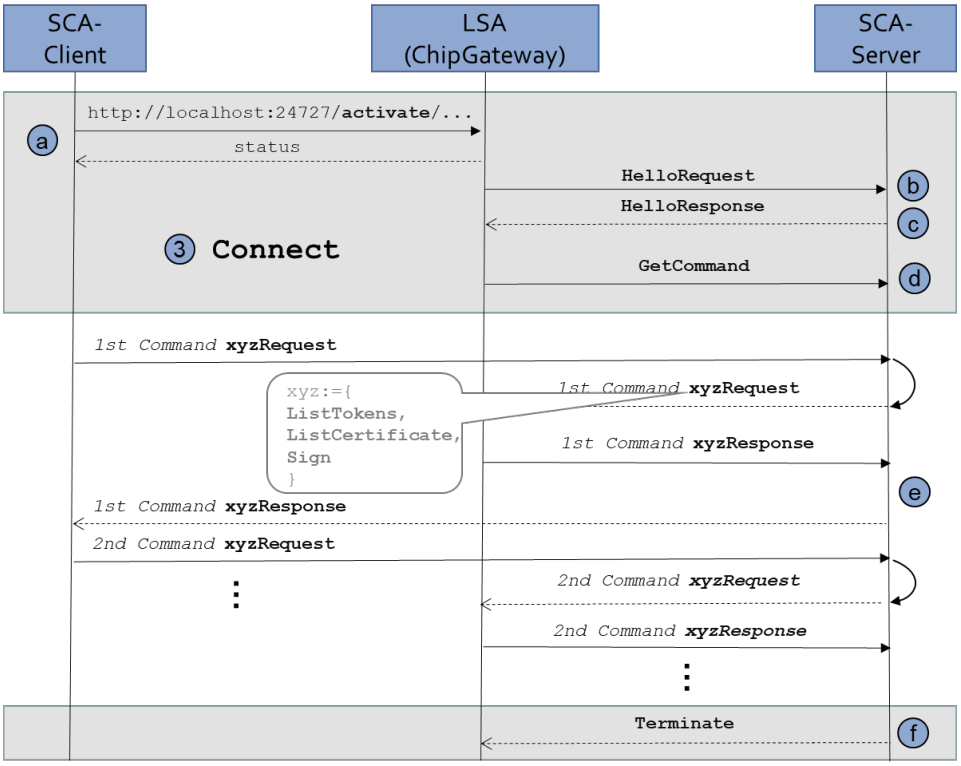

Fig. 2: Connection establishment within the ChipGateway protocol

As depicted in Fig. 2 the connection establishment process consists of the following steps:

a) The SCA Client starts the connection establishment by sending an appropriate http-`GET` request to the Local Signing Application (LSA), which listens at http://localhost:24727/activate. After the SCA-Client received the status, it may

[4] See [ET16a], [ET16b] and [ET16c].

involve the Signer and then send appropriate commands such as a `ListTokensRequest` to the SCA Server in phase (e), which can forward it to the LSA as soon as the connection is established after receiving `GetCommand` in phase (d).

b) Now the LSA establishes a TLS-protected connection to the SCA-Server at the `ServerAddress` and sends a `HelloRequest`, which among other parameters contains a `Challenge`.
c) The SCA Server answers with a `HelloResponse`, which among other parameters contains a `Signature` for the provided `Challenge` to advance the connection establishment.
d) The LSA sends a `GetCommand`, which essentially asks the SCA Server for the first command, while transporting information with respect to locally connected signature creation devices to the SCA Server within a `TokenInfo` component.
e) The set of commands, which may be sent from the SCA Client via the SCA Server to the LSA comprise `ListTokensRequest`, `ListCertificatesRequest` and `SignRequest` (see phases (3) and (4) outlined above).
f) Finally the SCA Server is able to terminate the connection by sending a `Terminate` message.

2.2 Remote Signing

The standardised protocol [DR07] supporting a broad range of digital signature services exists since 2007 and has been tailored by various profiles[5] and complemented by extensions, such as [NC15] for example. While the initial version of this family of standards was exclusively based on XML, the current revision [KH18b] also supports JSON syntax. Specific aspects relevant for the eIDAS-Regulation are addressed in [KH18a].

A set of JSON and REST based APIs for remote signature generation has been developed by the "Cloud Signature Consortium" in [CS18]. The specification is currently available as "preliminary release" and contains the following operational[6] functions:

- `info` – returns information on the remote service[7] and the list of API methods it has implemented.
- `auth/login` – authorises the remote service with HTTP Basic or Digest authentication.

[5] See https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dss and https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=dss-x .
[6] In addition to these operational endpoints [CS18] (Table 3) also lists the optional [HA12] specific endpoints `oauth2/authorize`, `oauth2/token` and `oauth2/revoke` for the initiation of an OAuth 2.0 based authorization flow, the issuance of access tokens or refresh tokens and the possible revocation of OAuth tokens respectively.
[7] The „remote service" in [CS18] is the „Server Signing Application" (SSA) according to [EN18b] and [EN18c].

- `auth/revoke` – revokes the service access token or refresh token.
- `credentials/list` – returns the list of credentials associated to a user.
- `credentials/info` – returns information on a signing credential, its associated certificate and a description of the supported authorisation mechanism.
- `credentials/authorize` – authorises the access to the credential for signing.
- `credentials/extendTransaction` – extends the validity of a multi-signature transaction.
- `credentials/sendOTP` – starts the online OTP mechanism associated to a credential.
- `signatures/signHash` – calculate a raw digital signature from one or more hash values.
- `signatures/timestamp` – return a time stamp token for the input hash value.

Last, but not least, the currently emerging ETSI standard [TS18] for remote signature generation contains XML and JSON profiles, which are based on [KH18b] and [CS18] respectively.

3 How to harmonise local and remote signing

3.1 Generic system architecture for local and remote signing

Comparing the existing interfaces and protocols for local and remote signing outlined in Section 2, it becomes obvious that the corresponding system architectures can easily be harmonised. The key aspect is that on a sufficiently high level of abstraction, there is no major difference between local and remote signing and the four phases of the ChipGateway protocol (`Init`, `Connect`, `List`, `Sign`) described in Section 2.1 are also existing in the generic and remote signature case as outlined in Fig. 3 and Fig. 4 respectively. The obvious differences between local and remote signing are in the `Connect` phase and the activation of the QSCD, which in the local case typically consists of entering a PIN, while in the remote case there needs to be a more sophisticated "Signature Activation Protocol" (SAP), which fulfils the requirements specified in [EN18b] (SRA_SAP).

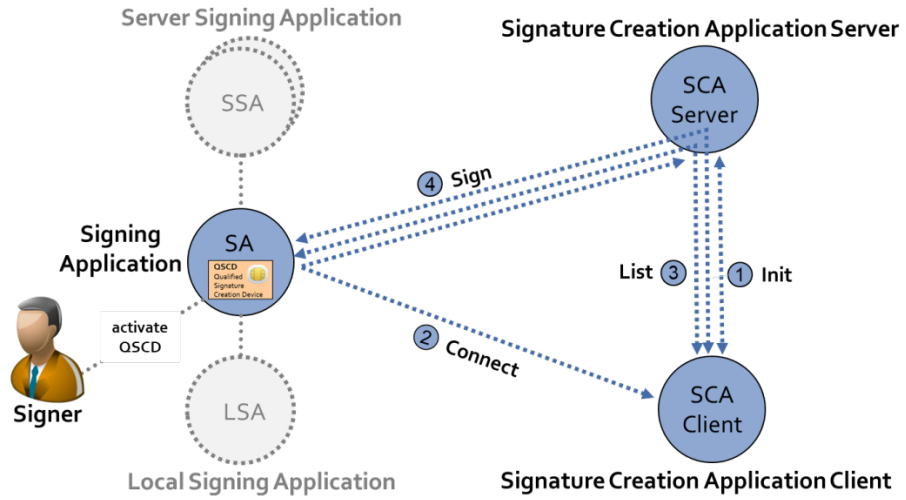

Fig. 3: Harmonised generic system architecture for local and remote signing

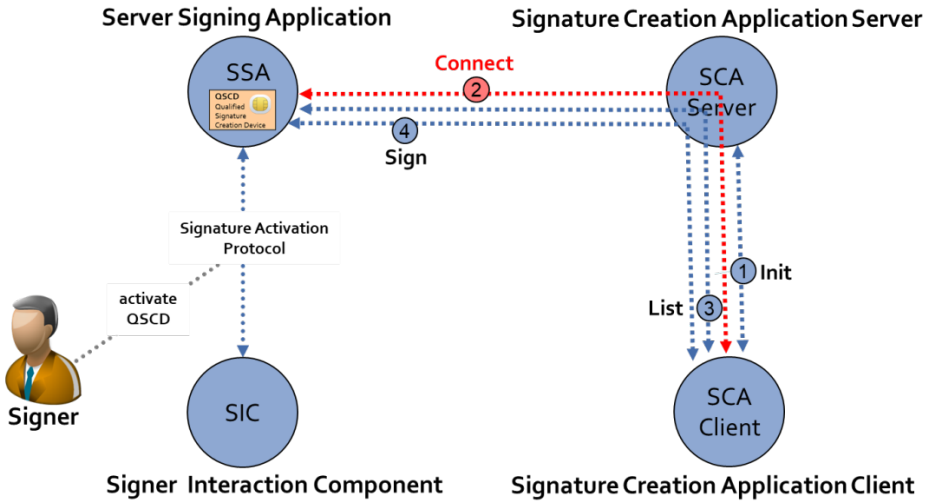

Fig. 4: System architecture for remote signing

3.2 Necessary changes for harmonisation of local and remote signing

Considering the details of the existing ChipGateway protocol [LE17] the following four changes to the [LE17] specification are required:

1. Enhanced `SigningApplication` structure instead of simple `ServerAddress`

   Whereas the ChipGateway protocol simply returns the ChipGateway endpoint of the SCA Server (`ServerAddress`) in the `Init` phase, there needs to be an enhanced `SigningApplication` structure, which will be present for all (local and server) signing applications which are available for a specific user. This structure shall contain a URI (`SigningApplicationIdentifier`) and may contain a subset of the information provided by an `info` endpoint according to [CS18].

2. User accounts for Signers at Server Signing Application

   Unlike in the local case, there needs to be a user account for the Signer at each involved Server Signing Application. The involved authentication procedures may be separated from the rest of the remote signing protocol using the generic mechanism defined in [FR14] together with the registered HTTP Authentication Schemes[8], which in particular comprise the use of OAuth 2.0 bearer tokens according to [JH12]. If there may be more than one Server Signing Application, it is advisable to use some sort of Single Sign-On (SSO) mechanism using standardised protocols for this purpose such as [CK05] or [SB14] for example, which may be combined with the bearer token usage according to [JH12]. Note, that a suitable SSO mechanism may not only be used for authentication purposes, but for "smart enrolment", as explained in Section 4.

3. `TokenInfo` needs to contain `SigningApplicationIdentifier`

   The `TokenInfo` structure, which is contained in `ListTokensResponse` for example, needs to contain the `SigningApplicationIdentifier` (see 1. above).

4. `PIN` based QSCD activation needs to be generalised to support suitable SAPs

   While the `SignRequest` within ChipGateway [LE17] contains the optional parameter `PIN`, which may contain the encrypted PIN, this aspect needs to be generalised in order to support suitable Signature Activation Protocols (SAPs), which fulfil the requirements defined in [EN18b].

4 Smart enrolment for remote signing

According to Art. 24 (1) of the eIDAS-Regulation [EU14], there are different options for the identification of the subject for the enrolment for qualified certificates:

a) "by the **physical presence** of the natural person or of an authorised representative of the legal person; or

[8] See http://www.iana.org/assignments/http-authschemes .

b) remotely, using **electronic identification means**, for which prior to the issuance of the qualified certificate, a physical presence of the natural person or of an authorised representative of the legal person was ensured and which meets the requirements set out in Article 8 with regard to the assurance levels 'substantial' or 'high'; or

c) by means of a **certificate** of a qualified electronic signature or of a qualified electronic seal issued in compliance with point (a) or (b); or

d) by using **other identification methods recognised at national level** which provide equivalent assurance in terms of reliability to physical presence. The equivalent assurance shall be confirmed by a conformity assessment body."

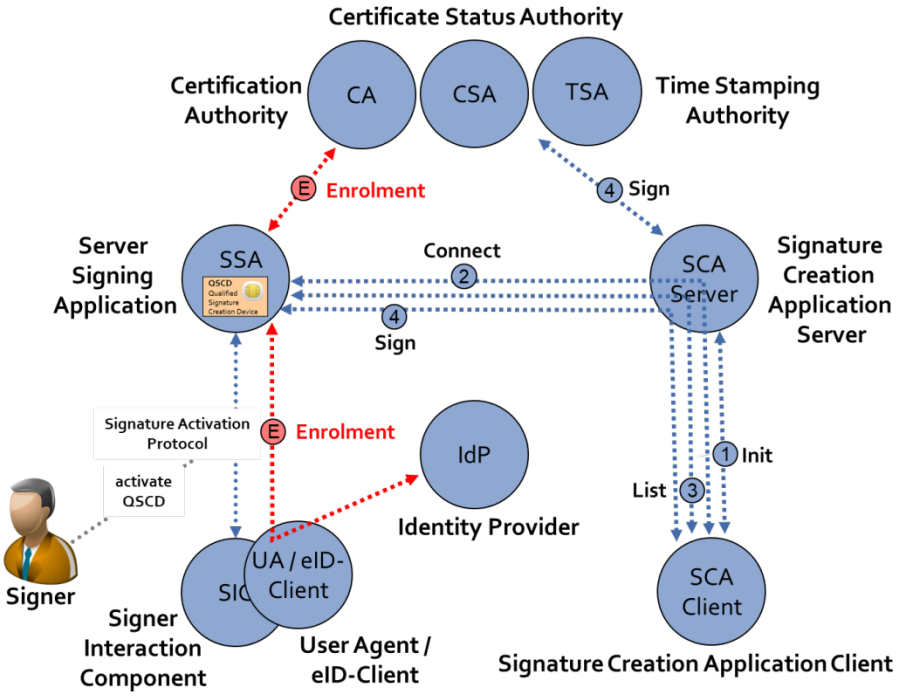

Fig. 5: Remote signing architecture with smart certificate enrolment

Against this background, the remote signing architecture outlined in Fig. 4 can easily be extended to the one inFig. 5, which also supports eID-based enrolment according to Art. 24 (1) (b) of the eIDAS-Regulation [EU14], if both the Signer and the Identity Provider support a suitable eID-scheme. For this purpose the Server Signing Application acts as Service Provider according to [CK05] or Relying Party according to [SB14], which redirects the Signer to the Identity Provider and receives later on a signed assertion containing the identity attributes of the Signer, which form the basis for the certificate

enrolment involving the Certification Authority[9].

In a similar manner, the system architecture outlined in Fig. 5 can be used for smart certificate enrolment according to Art. 24 (1) (c) and (d), whereas the Signer is not redirected to an Identity Provider, as in the eID-case (b) outlined above, but to a suitable signing service (e.g. the Signature Creation Application) for case (c) and a corresponding video-identification service for example for case (d).

5 Summary and Outlook

In the present contribution we have shown in Section 3 that it is easy to harmonise local and remote signing by slightly generalising the [LE17] protocol.

Furthermore we have outlined in Section 4 how the proposed authentication strategy based on [FR14], [JH12], [CK05] and [SB14] gives rise for "smart enrolment" procedures in line with Art. 24 (1) of the eIDAS-Regulation [EU14].

To facilitate the practical application of the ideas sketched in the present document, it may be worthwhile to standardise them within suitable technical committees of pertinent standardisation organisations, such as OASIS DSS-X and ETSI ESI for example. A first step in this direction is the recently started work on [HN18].

Bibliography

[TR15a] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik, BSI), „Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS token", Technical Guideline Nr. 03110, Part 1-4, <https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr03110/index_htm.html>

[TR15b] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik, BSI), „eCard-API-Framework", Technical Guideline Nr. 03112, Part 1-7, <https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr03112/index_htm.html>

[TR17] Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik, BSI), "eID-Client", Technical Guideline Nr. 03124, Part 1-2, <https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr03124/index_htm.html>

[LE17] LuxTrust S.A., ecsec GmbH, "ChipGateway Protocol – OASIS

[9] Note, that the Certificate Status Authority and the Time Stamping Authority only have been added in Fig. 5 for the sake of completeness. These components could have been added to Figure 1, 3 and 4 as well, as they are involved within the `Sign` phase to build appropriate AdES formats using certificate revocation information and time-stamp tokens, if required.

Contribution", <https://www.oasisopen.org/committees/download.php/60049/ChipGateway-Specification-OASIS.pdf>.

[EU15] Commission Implementing Decision (EU) 2015/296 of 24 February 2015 establishing procedural arrangements for cooperation between Member States on electronic identification pursuant to Article 12(7) of Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market, <http://data.europa.eu/eli/dec_impl/2015/296/oj>

[CS18] Cloud Signature Consortium, "Architectures and Protocols for Remote Signature applications", Public pre-release version 1.0.2.4 rev. PR (2018-09), < https://cloudsignatureconsortium.org/wp-content/uploads/2018/09/CSC_API_v1_PR_1.0.2.4.pdf>

[EU14] Regulation (EU) No 910/2014 of the European Parliament and of the Council of of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC, <http://data.europa.eu/eli/reg/2014/910/oj>.

[ET16a] EN 319 122, "Electronic Signatures and Infrastructures (ESI); CAdES digital signatures", Part 1-2

[ET16b] EN 319 132, "Electronic Signatures and Infrastructures (ESI); XAdES digital signatures", Part 1-2

[ET16c] EN 319 142, "Electronic Signatures and Infrastructures (ESI); PAdES digital signatures", Part 1-2

[EN14] EN 419 211, "Protection profiles for secure signature creation device", Part 1-6.

[EN18a] EN 419 221-5, "Protection profiles for Trust Service Provider Cryptographic modules – Part 5: Cryptographic Module for Trust Services", 2018.

[EN18b] EN 419 241-1, "Trustworthy Systems Supporting Server Signing – Part 1: General System Security Requirements", European Norm, 2018.

[EN18c] EN 419 241-2, "Trustworthy Systems Supporting Server Signing – Part 2: Protection profile for QSCD for Server Signing", European Norm, 2018.

[HÜ05] Hühnlein, D., „Die CCES-Signature-API – Eine offene Programmierschnittstelle für langfristig beweiskräftige elektronische Signaturen", in Proceedings of „Sicherheit 2005", LNI 62, Pages 361-374, 2005, <http://www.ecsec.de/pub/2005_Sicherheit.pdf>

[IS14] ISO/IEC 24727, "Identification cards – Integrated circuit card programming interfaces", Part 1-6, 2014.

[MS18] Microsoft, "CryptoAPI System Architecture", <https://docs.microsoft.com/en-us/windows/desktop/seccrypto/cryptoapi-system-architecture>.

[KH18a] Kuehne, A., Hagen, S., "Advanced Electronic Signature Profile for OASIS Digital Signature Services Version 2.0", Working Draft

[HN18] Hühnlein, D., von Nigtevecht, E. J., "Local and Remote Signature Profile for OASIS Digital Signature Services Version 2.0", Working Draft

[NC15] von Nigtevecht, E. J., Cornelis, F., „DSS Extension for Local Signature Computation Version 1.0", Committee Specification 01, 27 July 2015, <http://docs.oasis-open.org/dss-x/localsig/v1.0/cs01/localsig-v1.0-cs01.pdf>

[DR07] Drees, S., "Digital Signature Service Core Protocols, Elements, and Bindings Version 1.0", OASIS Standard, 11 April 2007, <http://docs.oasis-open.org/dss/v1.0/oasis-dss-core-spec-v1.0-os.html>

[KH18b] A. Kuehne, S. Hagen, "Digital Signature Service Core Protocols, Elements, and Bindings Version 2.0", Committee Specification Draft 01 / Public Review Draft 01, 29 August 2018, <http://docs.oasis-open.org/dss-x/dss-core/v2.0/csprd01/dss-core-v2.0-csprd01.pdf>

[SB14] *OpenID Connect Core 1.0*., edited by N. Sakimura, J. Bradley, M. Jones, B. de Medeiros and C. Mortimore, 8 November, 2014, <http://openid.net/specs/openid-connect-core-1_0.html>.

[EI18] Opinion No. 3/2018 of the Cooperation Network on the Luxemburgish eID scheme, <https://ec.europa.eu/cefdigital/wiki/pages/viewpage.action?pageId=65972753>

[PC13] PC/SC Workgroup, "PC/SC Workgroup Specifications", <https://www.pcscworkgroup.com/specifications/>.

[GF15] Griffin, R. and Fenwick V., "PKCS #11 Cryptographic Token Interface Base Specification Version 2.40", OASIS Standard, 14 April 2015, <http://docs.oasis-open.org/pkcs11/pkcs11-base/v2.40/os/pkcs11-base-v2.40-os.pdf>.

[HA12] Hardt D. (Ed.), "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>

[JH12] Jones M., Hardt D., "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, October 2012, <https://www.rfc-editor.org/info/rfc6750>

[FR14] Fielding, R., Reschke, J. (Eds.): "Hypertext Transfer Protocol (HTTP/1.1): Authentication", June 2014, <https://www.rfc-editor.org/info/rfc7235>

[CK05] *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*. Edited by Scott Cantor, John Kemp, Rob Philpott and Eve Maler, 15 March 2005. OASIS Standard <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>.

[TS18] ETSI, "Electronic Signatures and Infrastructures (ESI); Protocols for remote digital signature creation", Draft ETSI TS 119 432, V0.0.5 (2018-07).