D3.4

Discovery of Trust Scheme Publication Authorities

Document Identification	
Date	30.05.2018
Status	Final
Version	Version 1.0

Related WP	WP 3	**Related Deliverable(s)**	D3.3
Lead Authors	FHG	**Dissemination Level**	PU
Lead Participants	FHG	**Contributors**	USTUTT, NLNET, ATOS, TUG, GS
Reviewers	TUBITAK		

This document is issued within the frame and for the purpose of the LIGHT*est* project. LIGHT*est* has received funding from the European Union's Horizon 2020 research and innovation programme under G.A. No 700321.

1. Executive Summary

This document provides discovery mechanisms for Trust Scheme Publication Authorities in order to be able to locate the right trust data in the global name space of domain names. This is one part of the Trust Scheme Publication Authority (TSPA), which provides the infrastructure for the Publication of Trust Schemes as well as for the Discovery and Verification of Trust Scheme Memberships. For this purpose, the conceptual framework of the TSPA was developed in D3.1, which consists of two components; a DNS Name Server with DNSSEC extension and a Trust Scheme Provider. The design of the Publication of Trust Schemes is described in D3.3.

Within LIGHT*est*, discovery of trust-related information is not only required for Trust Scheme Publication Authorities, but also for Trust Translation Authorities and Delegations. This is very similar to the Publication of trust-related information. For the publication of trust-related information for trust schemes, a consolidated approach was developed (see D3.3, D4.3, and D5.3), which also includes already sections for the discovery of trust declarations.

In this deliverable, the consolidated approach is specified now for the discovery mechanisms for Trust Scheme Publication Authorities. This includes the discovery of the trust scheme that the trust service provider claims to be a member of (issue DNS queries), the verification process of the Trust Scheme Membership (checking trust list) and the authenticity of the trust list (limit accepted certificates using DNS SMIMEA record type).

In addition, information how to discover the policy and rules of the corresponding trust scheme, which can be found in the scheme information section of the signed trust list, are provided following the ETSI TS 119 612 [1] standard.

Furthermore, possibilities and examples of direct pointers to the relevant DNS data embedded in X509.signatures, SAML assertions, and signed documents are presented. With the OpenID Connect [2] protocol, a complementary mechanism for the discovery mechanisms for Trust Scheme Publication Authorities is also outlined.

By way of example, the applicability of the consolidated approach for the discovery of Trust Scheme Publication Authorities is demonstrated. This also includes the verification of Trust Scheme Memberships and the validity of the certificate used for signing the trust list. Therefore, the examples for eIDAS Germany and the Electronic Signature Law of the People's Republic of China of D3.3 are revisited and extended accordingly. Hence, these examples show the complete chain for the association of an Issuer with a Trust Scheme including the discovery and verification of the Trust Scheme Membership as well as the verification of the authenticity of the trust list.

2. Document Information

2.1 Contributors

Name	Partner
Heiko Roßnagel	FHG
Sven Wagner	USTUTT
Martin Hoffmann	NLNET
Olamide Omolola	TUG
Georg Wagner	TUG
Jesse Kurtto	GS
Javier Presa	ATOS
Miguel A. Mateo	ATOS
Miryam Villegas	ATOS

2.2 History

Version	Date	Author	Changes
V0.1	16.3.2018	FHG	Initial draft, Table of Content
V0.2	25.04.2018	FHG, USTUTT	Added Sections 6.1, 6.2
V0.3	26.04.2018	NLNET	Added Chapter 5
V0.4		NLET, USTUTT	Added Section 6.3 and 9.1
V0.5	03.05.2018	GS	Added Section 7
V0.6	07.05.2018	TUG	Added Section 9.2
V0.7	08.05.2018	ATOS	Added Section 8
V0.8	09.05.2018	FHG, USTUTT	Compilation of final draft
V0.9	23.05.2018	TUBITAK	Internal Review
V1.0	30.05.2018	FHG	Integrating review suggestions, formatting changes

3. Table of Contents

3.1 Table of Figures

3.2 Table of Tables

3.3 Table of Acronyms

ATV	Automatic Trust Verifier
CAA	Certification Authority Authorization
DANE	DNS-based Authentication of Named Entities
DNS	Domain Name System
DNSSEC	Domain Name System SECurity extensions
eIDAS	Electronic IDentification And Signature (Regulation (EU) No 910/2014 on electronic identification (eID) and trust services for electronic transactions in the internal market)
LoA	Level of Assurance
NAPTR	Name Authority Pointer
SAML	Security Assertion Markup Language
SAN	Subject Alternative Names
TL	Trusted List
TLS	Transport Layer Security
TSL	Trust-Service Status List
TSPA	Trust Scheme Publication Authority
URI	Uniform Resource Identifier

4. Scope of the Deliverable

This deliverable D3.4 describes the discovery of Trust Scheme Publication Authorities in the global namespace of domain names. It builds on the results of the deliverables D3.1 “Conceptual Framework for Trust Schemes” and D3.3 “DNS-based Publication of Trust Schemes”. In addition to the discovery itself, the verification of the trust scheme memberships as well as the verification of the validity of the certificate used for signing the trust lists are important tasks in this process. Furthermore, it might be relevant to discover information on the policy of the trust scheme, the trust service provider claims to be a member of. Therefore, the topics covered in this deliverable are the following:

In Section 5, the consolidated approach to publishing Trust-related Information in the DNS in general is presented. This approach was developed for the publication of trust-related information for trust schemes, trust translation lists and trust delegations and it was published already in the corresponding deliverables D3.3, D4.3, and D5.3. However, in addition to the publication of trust-related Information, it also includes sections for the discovery of trust declarations as well as the authenticity of Trust Declarations, which are relevant for this deliverable. This Section 5 is almost identical to the corresponding sections in the deliverables D3.3, D4.3, D5.3, D4.4, and D5.4. This leads to some duplication in the deliverables, however it provides for the readers standalone and complete documents.

In Section 6, the Discovery of Trust Schemes Memberships is presented. This includes a brief summary of the concept of the Trust Scheme Publication Authority (TSPA) in D3.1 with a special focus on the issues relevant to Discovering Trust Scheme Association (Section 6.1). The information flow for the discovery and verification process of Trust Scheme Memberships is shown in Section 6.2. Furthermore, relevant tags, which provide information and links for the policy and rules of the corresponding trust scheme are presented in Section 6.3.

Pointers to relevant DNS data are listed in Section 7. This includes pointers in X.509 certificates, SAML assertions, as well as signed documents.

In Section 8, a complementary mechanism to discover trust schemes is presented using the OpenID Connect protocol.

The applicability of the consolidated approach to discover Trust Scheme Publication Authorities, verify trust scheme memberships as well as the validity of the certificate used for signing the trust list is demonstrated in Section 9. In that section, two examples of D3.3, namely eIDAS Germany and the Electronic Signature Law of the People’s Republic of China are revisited and extended in particular demonstrating the authenticity of trust declarations.

5. A Consolidated Approach to Publishing Trust-related Information in the DNS

The trust framework created by LIGHTest verifies the trustworthiness of an electronic transaction by attempting to establish a chain of trust from a set of pre-configured, well-known trust sources to this transaction. The links in this chain are assurances that the trust into a source can be extended to another entity. This section looks at the basic properties these assurances exhibit independently of their concrete contents and introduces an underlying, fundamental framework.

Within the LIGHTest project, three working packages look at different aspects of such assurances. WP3 examines the most basic form where a trust source known as a trust scheme declares that some issuer of trust services, such as certificates or time stamps, conforms to the conditions and rules set out by the scheme and thus extends trust placed into it onto this trust service. WP4 assesses the relationship between multiple trust schemes allowing a trust scheme or some other trusted entity to declare if and how trust into one scheme extends to trust into another scheme. WP5, finally, looks into how individual entities can empower other entities to act on their behalf– extending trust into themselves onto that other entity within certain well-defined limits.

5.1 Trust Declarations

However different they may appear, each of these aspects follows a similar pattern: some entity makes a *trust declaration* stating that trust into a certain entity extends to another entity, possibly providing conditions and limits of such an extension of trust. To simplify further discussion, it will be helpful to label the three entities involved in the process. The entity issuing the declaration shall be the *originator,* the entity that is already trusted is the *source* and the entity trusted as a result of the declaration is the *target.*

Within the aspects discussed as part of the LIGHTest project, in many cases the originator of a declaration is identical to the source. This is certainly true for trust membership publication in WP3, where the trust scheme itself declares which trust services are a member. In the aspects of the other two work packages, similarly originator and source are most often identical. However, there may be use cases where this is not the case. For instance, a third party may declare a trust translation independently of the trust schemes that are source or target of this declaration. Similarly, a third party may declare a trust delegation. For instance, business registries often provide information about individuals that are allowed to sign on behalf of a company. Such information can be modeled as third-party trust delegation.

This definition specifies a single declaration to extend trust from exactly one source to exactly one target. In practice, the document formats used often contain multiple declarations according to this definition. For instance, the trusted lists defined in ETSI TS 119 612 [1] used for the declarations in WP3 contain a list of all the targets that a trust scheme as a source wishes to

extend trust to. This published form of declarations shall be called *trust declaration documents.* At least in principle each such document can contain one or many declarations with any number of sources and targets.

For a trust declaration to be considered when building the chain of trust during validation of an electronic transaction, the declaration itself needs to be trusted. If a declaration document is treated like any other electronic transaction, this trust can in turn be established through verification using the LIGHTest framework. That is, there needs to be a chain of trust from pre-configured trust sources to the originator of the declaration document for the particular aspect of the declaration in question.

Note that in most cases where the originator of the declaration is identical to the source of the declaration, this chain exists implicitly and no extra checks are needed. It may, however, be possible that conditions for trusting the source as such and trusting declarations made by it are different and thus need to be verified independently.

5.2 Publication of Trust Declarations

When using trust declarations for verifying an electronic transaction, a validator needs to construct a chain of trust declarations leading from any of the trusted entities to those entities appearing in the transaction. It does so by recursively finding and adding applicable trust declarations that have an originator that is either a trusted entity or the target of the declaration is already part of the chain. In order to do this in an unaided, automatic way, the validator needs a way to gain access to all declarations that are potentially usable in this process.

There are two fundamental strategies for the verifier to find declarations when they are needed: a declaration could either be actively supplied as part of the input or configuration or it is left to the verifier itself to discover it.

The prime example for the former case is that a declaration is provided as part of the electronic transaction to be verified. This is particularly useful if the creator of the transaction is aware that the declaration is necessary for verification. For instance, if an entity signs a document in their function as a proxy, the transaction will only ever verify if the declaration of trust delegation – the mandate – is known to the verifier. The proxy may very well include the mandate in the transaction right away.

For declarations that are potentially applicable to a large amount of transactions or if the sender of the transaction does not know the trusted entities the receiver will base their verification on, such a strategy isn't very practical. Instead, it is better to make the declarations available publicly and provide means for a verifier to discover how and where it can retrieve them. The verifier can then decide itself which declarations it needs and try and find them as needed.

While the most likely method for publishing declarations currently is the Hypertext Transfer Protocol (HTTP), other protocols may become available in the future. An extensible standard exists to describe both the method used for accessing a resource and all necessary parameters for a successful retrieval in the form of a Uniform Resource Identifier (URI). It encodes all information into a single string which can be easily stored or transmitted.

5.3 Discovering Trust Declarations

In order to build a chain of trust from published declarations, the verifier needs to be able to discover their existence. Given a transaction and a set of already trusted entities, this chain can be built from two sides: either the verifier starts with a trusted entity, tries to discover all the declarations that have this entity as their source, and repeats this process until it arrives at a declaration that includes the transactions as its target or runs out of declarations to apply. Alternatively, it can start with the transaction, attempts to discover all declarations that have the transaction as their target, and continues by recursively trying to find declarations that have the sources of already discovered declarations as their target until it arrives at an already trusted entity as a declaration's source or, again, runs out of declarations.

In both cases, the verifier needs a mechanism to search for the URI of a declaration based on a given entity. Such a mechanism will have to take some information that identifies that entity as input. Given that entities are typically identified by X.509 certificates, it should be possible to include such identifying information in the entity's certificate. One of the possible options is to use a domain name as the entity's identifier. This has the advantage that the name can be used directly as a search input for the DNS, allowing to use the DNS as a global, highly available, distributed, and independently managed data store.

A domain name can be stored in a certificate either in the subject alternative name or issuer alternative name extension. The subject alternative name indicates the domain name identifying the entity using the certificate while the issuer alternative name identifies the entity having issued the certificate.

Using this domain name as input, the URIs to the declarations for differing aspects should be stored in the DNS. As the DNS uses record types to distinguish between different types of information stored for a domain name, one option is to register an individual record type for each aspect of trust declarations. However, this would clutter the space of types. As the data stored is the same in every case – a URI – an alternative approach can be used whereby the aspect is encoded as a prefix to the domain name. This is already used for instance with the SRV record type for discovering the host name and port where a certain networking service is available for a domain. As such services sometimes are available over different transport protocols, a two-layer prefix of the form *_service._protocol* is used where the latter describes the transport protocol to be used and the former the networking service.

In keeping with this concept, LIGHTest proposes to use the DNS for providing pointers to all kinds of declarations via a pair of prefixes of the form *_aspect._application*. Here, the type of application for which a declaration is published is the second part of the prefix while the first part defines the particular aspect within that application. For LIGHTest itself, the application is, of course, trust-related declarations, which is identified via using the literal label *_trust* as the second part. Each of the aspects of trust declarations defines its own label to be used as the first part.

Under the name constructed by concatenating the prefix with the entity's own identifying domain name, the entity can now publish pointers to trust declarations of the corresponding aspect that relate to it. It can use the already existing URI resource record type for this purpose. This type is defined in section 4 of RFC 7553 [3]. Its record data contains exactly one URI. While section 5 of the RFC describes a different use of the record, this use is limited by a different set of prefixes, allowing its reuse for declaration publication based on the prefixes defined above.

If the entity in question is not the originator of a declaration it may not control the URI under which the declaration is published. In this case, it may be beneficial to only point to the originating entity rather than burden itself with tracking whether the originators URI has changed. This can be done easily if the originator is an entity identified by a domain name, too. In this case, instead of publishing URI resource records under its domain name prefixed by the declaration aspect, the entity will publish PTR resource records. Such record types, part of the original DNS specification in RFC 1035 [4], contain another domain name as their record data. If such records are present, they instruct the verifier to continue discovery for declaration at the entity identified by these domain names. Note that as these names in the PTR record's data refer to the specific application and aspect and as such are already prefixed with the correct declaration prefix. While not used in the LIGHTest framework as yet, this would allow to provide references between different aspects.[1]

5.4 Authenticity of Trust Declarations

If a verifier retrieves a declaration from somewhere in the network, it needs to make sure that the data it received is indeed the declaration made by the originator. Since the URI resource records are stored under the domain name identifying the originator, it is reasonable to assume they are authentic if DNSSEC validation succeeds, guaranteeing that the URI is indeed the one intended by the originator.

[1] The initial version of the Consolidated Framework as presented in deliverable D3.3 proposed to let the PTR records refer to the domain name identifying the entity. After discussion, it was decided that the updated approach presented here is both more correct as it points to the exact name where discovery continues and, as mentioned, more flexible as a generic means of discovery.

In the next step, where the verifier contacts the server indicated in the URI, it needs to ensure that it communicates with the correct server. When using encrypted transport via the TLS protocol, the server will identify itself via a certificate. In order to deal with shortcomings of the traditional method of certificate verification, a protocol called DNS-based Authentication of Named Entities or DANE for short allows a server operator to publish information about the certificates used in DNS.

Since, however, the server is not necessarily operated under authority of the originator of the declaration – for instance because the declarations are hosted by a third party that provides better availability, this does not guarantee that the declaration received is indeed the one that the originator intended.

This final link can be provided if the declaration itself is a signed document. The originator can then publish the certificates that it uses for signing declarations of a certain aspect using a slightly adapted version of the DANE protocol.

To do so, it adds SMIMEA resource records under the same domain name it placed the URI records pointing to the declarations. These records define conditions a certificate has to fulfill to be accepted. By placing these records, the originator declares that all documents retrieved via the pointers have to verify considering these conditions.

6. Discovery of Trust Schemes Memberships

6.1 Concept for Discovering Trust Scheme Association

The overall concept for Discovering Trust Scheme Association was already developed in D3.1 [5]. This section provides a brief summary of this concept with a special focus on the issues relevant to Discovering Trust Scheme Association. In addition, the concept was further refined since then and this requires some modifications in the resource records stored in DNS.

The role of the Trust Scheme Publication Authority (TSPA) in LIGHTest is to provide the infrastructure for the discovery and verification of trust scheme memberships. Therefore, the concept of the TSPA consists first of a DNS Name Server (with DNSSEC extension) which enables the discovery of associated Trust Scheme and Trust Scheme Provider of a certificate Issuer. The second component is the Trust Scheme Provider, which provides a signed Trust List indicating that a certificate Issuer is trusted and operates under the trust scheme of the Trust Scheme Provider. In addition, the Trust Scheme Provider contains the Tuple-Based representation of a Trust Scheme. This is illustrated in Figure 1, which is an updated version of Figure 4 in D3.1 [5] and also of Figure 1 in D3.3 [6]. It depicts the DNS Name server with data containers for the Certificate Issuer, for the trust scheme name (for boolean trust schemes) or for each ordinal level of a trust scheme (for ordinal trust schemes), as well as the Trust Scheme Provider with the signed Trust List and Tuple-Based representation of a Trust Scheme.

In the resource records of the DNS Name Server, the data containers for the Issuer, trust scheme name and ordinal level of a trust scheme do not include the fingerprints of the corresponding certificates of the issuer and trust scheme provider anymore. Instead, the certificates, which are accepted for signing the trust list, are limited. This can be realized by implementing certificate constraints in the DNS using the SMIMEA DNS resource record. These modifications in the resource records are a result from the consolidated approach to publishing trust-related information in the DNS (for further details see section 5).

The SMIMEA resource record was already introduced in the deliverable D2.7 “Relevant DNSSEC Concepts and Basic Building Blocks” [7] and it is defined in [8]. The SMIMEA mechanism associate an SMIME user's certificate with the intended domain name by a number of ways to limit the accepted certificates. This is very similar to DANE, where the TLSA mechanism limits the certificates used for TLS [9]. The format of the SMIMEA resource record is the same as for the TLSA record. It contains four fields (Certificate Usage, Selector, Matching Type, and Certificate Association Data) which can be used to constrain the certificates, which are accepted for signing the trust list.

Hence, in LIGHTest, the SMIMEA resource record is used to verify if the certificate used for signing the trust list is valid. This is demonstrated by way of example in section 9.

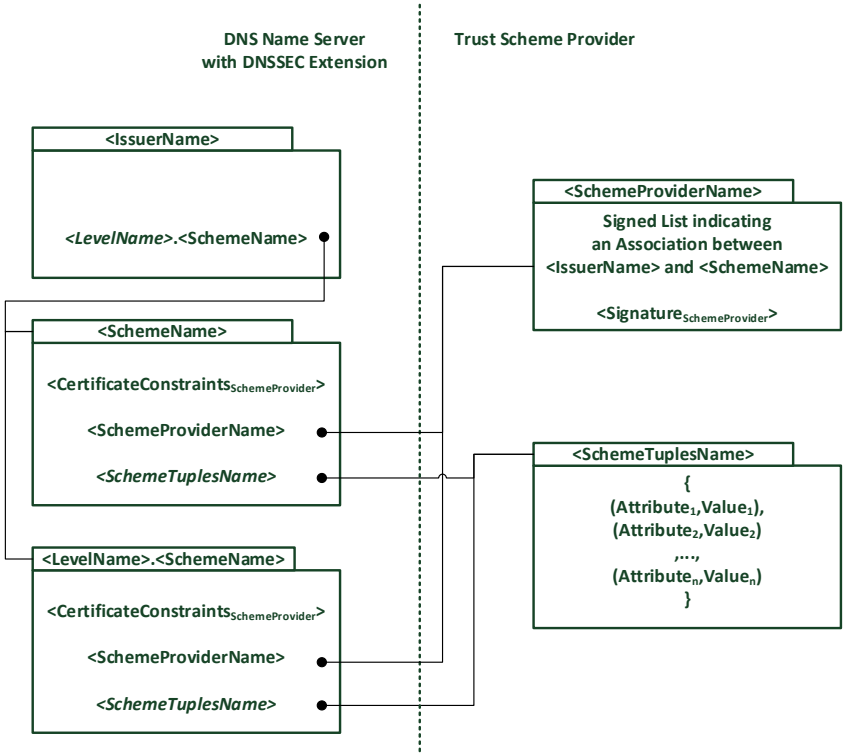

Figure 1 Overview on the updated concept for trust scheme publications in the TSPA

6.2 Querying of Trust Scheme Association

As described already in the LIGHTest Reference Architecture in D2.14 [10] and in D3.1 [5], the TSPA provides the capability of discovering and verifying Trust Scheme Memberships.

For example for the scenario for qualified electronic signatures and boolean trust schemes (see Figure 9 in [10]), the ATV extracts the signer certificate of the electronic transaction as well as the Issuer certificate (Step 2). After validation of the signatures (Step 3&4), the issuer name is extracted from the certificate (Step 5). The TSPA is now used for retrieving the associated trust scheme (Step 6) and verify if the Issuer operates under the specific Trust Scheme (Step 7).

These two steps (Step 6 and Step 7) from the reference architecture are described in the following in more details. In accordance to the TSPA Concept (see section 6.1), the querying of trust schemes in the TSPA also requires some modifications in the sequence diagrams as a result from the consolidated approach to publish trust-related information in the DNS in general. This is depicted in Figure 2, which is an updated version of Figure 7 in D3.1 [5].

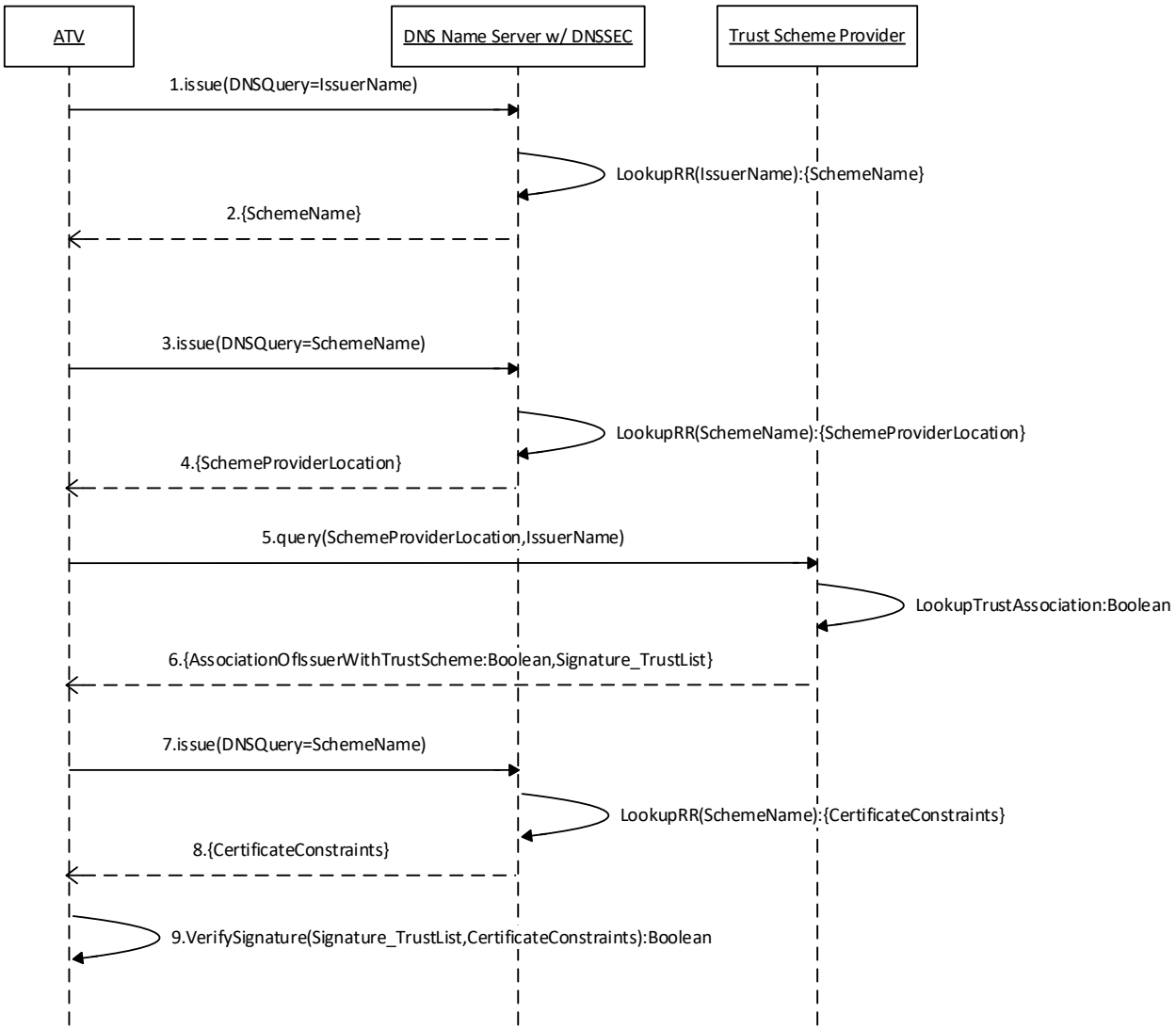

Figure 2: Querying of Trust Schemes in the TSPA

This process of querying trust schemes in the TSPA (see Figure 2) consists of the following steps:

1. The ATV issues a query to the DNS Name Server with the Issuer Name.
2. The DNS Name Server delivers the record for the Issuer Name that contains the name of the associated trust scheme. The name of the associated trust scheme indicates the Scheme Name (SchemeName), in the case of a Boolean Trust Scheme Publication, and the ordinal value of the Scheme Name (LevelName.SchemeName) in the case of an Ordinal Trust Scheme Publication.
3. The ATV now queries the DNS Name Server for the associated trust scheme.

4. The DNS Name Server provides the record for the Trust Scheme Provider, which contains the Location of the Trust Scheme Provider (SchemeProviderLocation).
5. The ATV now queries the Trust Scheme Provider by using SchemeProviderLocation, and provides the Issuer Name to the Trust Scheme Provider.
6. The Trust Scheme Provider queries its Trust List and verifies whether the Issuer Name is listed as operating under its Trust Scheme. The Trust Scheme Provider finally provides a signed statement of association of an Issuer Name with a Trust Scheme.
7. The ATV queries again the DNS Name Server for the associated trust scheme.
8. The DNS Name Server provides the SMIMEA record for the Scheme Name that contains constraints, which limit the accepted certificates for the signature of the trusted list (CertificateConstraints).
9. The ATV can now verify if the certificate used for signing the trusted list was valid.

In this process, the ATV verifies first that the statement of association of an Issuer with a Trust Scheme was made by the Trust Scheme Provider (Step6). Second, the ATV further verifies the validity of the certificate used for signing the trusted list (Step9).

In the Consolidated Approach to Publishing Trust-related Information in the DNS (see section 5) the message formats for the DNS queries and responses are defined. The specification of this consolidated approach for WP3 results in using Trust Membership Declarations where the trust scheme is both the source and originator of the declaration. Furthermore, the trusted list contains a list of all the targets the trust scheme wishes to extend trust to.

The prefix ***_scheme._trust*** used in DNS indicates within LIGHTest trust membership declarations and claims. Hence, the DNS entries are as follows:

A trust service provider points to the trust schemes it claims to be a member of by publishing PTR records with the trust scheme's domain name identifier under the ***_scheme._trust*** prefix.

A trust scheme publishes its trust membership declaration under the same ***_scheme._trust*** prefix using the URI record type. This URI points to the trust declaration document, the signed trust list.

The trust declaration document is a trusted list that indicates the membership of the trust service with the referred to trust scheme. For the trust declaration document existing standards on Trust Service Status Lists (e.g. ETSI TS 119 612 [1]) are used.

By way of example, the discovery and verification of trust scheme membership is demonstrated in section 9 for selected trust schemes.

To start the verification process, the ATV extracts the issuer name from the certificate (see above). Therefore, it is required that a trust service includes its own domain name identifier, its

certificates in the subject alternative name extension and, optionally, in the certificates issued in its role as a certificate issuer in the issuer alternative name extension. This is described in more detail in section 7.

6.3 Concept for Discovering Trust Scheme Policy

In addition to discovering and verifying Trust Scheme Memberships, which is described in the previous section, the discovery of the policy of the trust scheme, the trust service provider claims to be a member of, is another very important task.

In ETSI TS 119 612 [1], the trust scheme policy is specified as part of the Scheme information section in the overall structure of trusted lists. Hereby, several tags provide information on the trust scheme policy.

The tag <*SchemeTypeCommunityRules*> "specifies the URI(s) where users (relying parties) can obtain scheme type/community/rules information against which services included in the list are approved and assessed, and from which the type of scheme or community may be determined." Among other things "The referenced URI(s) shall identify (i) the specific policy/rules against which services included in the list are approved and assessed, and from which the type of scheme or community may be determined; (ii) the description about how to use and interpret the content of the trusted list."

The tag <*PolicyOrLegalNotice*> "specifies the scheme's policy or provides a notice concerning the legal status of the scheme or legal requirements met by the scheme for the jurisdiction in which the scheme is established and/or any constraints and conditions under which the TL is maintained and published. …Any referenced text shall provide information describing the policy under which the Scheme Operator operates or any relevant legal notices with which users of the TL should be aware."

The tag < *PointersToOtherTSL*> *"references any relevant trusted list or any list of trusted lists." It is mandatory for the trusted lists of the EU Member States and it includes the pointer to the List Of Trusted Lists (LOTL) of the EC.*

For the example for eIDAS Germany from the D3.3 "DNS-based Publication of Trust Schemes", which is also extended in this deliverable for the authenticity of trust declarations in Section 9.1, the entries of the tags, which provide information on the trust scheme policy (see above), are the following:

```
<SchemeTypeCommunityRules>
   <URI xml:lang="en">
   http://uri.etsi.org/TrstSvc/TrustedList/schemerules/EUcommon
   </URI>
   <URI xml:lang="en">
   http://uri.etsi.org/TrstSvc/TrustedList/schemerules/DE
```

```
      </URI>
   </SchemeTypeCommunityRules>


   <PolicyOrLegalNotice>
      <TSLLegalNotice xml:lang="en">
      The applicable legal framework for the present trusted list is
      Regulation (EU) No 910/2014 of the European Parliament and of the
      Council of 23 July 2014 on electronic identification and trust services
      for electronic transactions in the internal market and repealing
      Directive 1999/93/EC.
      </TSLLegalNotice>
      <TSLLegalNotice xml:lang="de">
      Der für diese Vertrauenslisten geltende Rechtsrahmen ist die Verordnung
      (EU) Nr. 910/2014 des Europäischen Parlaments und des Rates vom 23. Juli
      2014 über elektronische Identifizierung und Vertrauensdienste für
      elektronische Transaktionen im Binnenmarkt und zur Aufhebung der
      Richtlinie 1999/93/EG.
      </TSLLegalNotice>
   </PolicyOrLegalNotice>


   <PointersToOtherTSL>
      <OtherTSLPointer>
         <ServiceDigitalIdentities>...</ServiceDigitalIdentities>
         <TSLLocation>
         https://ec.europa.eu/information_society/policy/esignature/trusted-
         list/tl-mp.xml

         </TSLLocation>
         <AdditionalInformation>
            <OtherInformation>...</OtherInformation>
            <OtherInformation>...</OtherInformation>
            <OtherInformation>...</OtherInformation>
            <OtherInformation>...</OtherInformation>
            <OtherInformation>
               <SchemeTypeCommunityRules>
                  <URI xml:lang="en">
                  http://uri.etsi.org/TrstSvc/TrustedList/schemerules/EUlistof
                  thelists
                  </URI>
               </SchemeTypeCommunityRules>
            </OtherInformation>
         </AdditionalInformation>
      </OtherTSLPointer>
   </PointersToOtherTSL>
```

To conclude, for the discovery of the policy of the trust scheme, a verifier can search for and evaluate the content of the above listed and shortly described tags, which are provided in Trust Service Status Lists according to ETSI TS 119 612 [1]. This is not included in the standard ATV verification process for automatic trust verification. This is an optional task and

this section introduces how a verifier can discover the corresponding trust policy of the trust scheme.

7. Pointers to relevant DNS data

This section gives an overview on how direct pointers to the relevant DNS data can be added in different formats of electronic transactions. This includes pointers in X.509 certificates, in SAML assertions, and in signed documents.

7.1 Pointers in X.509 certificates

X.509 certificates are digital certificates, which are basically structured text files, formatted according to the standard RFC 5280 [11] The certificates can have one or more extensions embedded within.

Subject Alternative Name (SAN) is an extension to the X.509 specification that allows specifying more than one name in a single TLS certificate (RFC 5280, [11]). Unlike wildcard certificates, which can only include subdomains like *.example.com, a SAN certificate can include multiple domains like example.com and anotherdomain.com. A very common usage is to have a SAN certificate that includes the www. prefix in addition to the base domain, e.g. www.example.com and example.com.

The Issuer Alternative Name Extension (RFC 5280, [11]). adds a base-64 encoded blob that contains additional information about the issuer to the certificate. It consists of one or more attribute:parameter pairs that identify the issuing Certificate Authority (CA). Common attributes are DNSName (e.g. DNSName:ca.example.com) and X500Name (e.g. cn=Example CA, ou=EMEA Sales, o=Example CA, Ltd., c=DE).

7.1.1 Vulnerabilities

Even verifying the chain of trust all the way to a trusted root does not make DNS records pointed from X.509 certificates invulnerable to attack. As DNS records are not cryptographically protected, an attacker can acquire real but mis-issued TLS certificates by manipulating the relevant DNS servers and/or Internet routing tables. Having restricted DNS Certification Authority Authorization (CAA) records in place can help protect against these attacks, if DNS Security Extensions (DNSSEC) protects the domain zone itself.

7.2 Pointers in SAML assertions

SAML Service Provider (SP) and Identity Provider (IdP) form a mutual trust relationship by the exchange of cryptographically secured metadata. The issuer is typically referenced as an URL in XML format, e.g. <saml:Issuer>https://example.com/saml/metadata/examplesite</saml:Issuer>.

The metadata is typically exchanged manually, and DNS is only used to look up the IP address of the endpoint hosting the metadata. DNSSEC is not required, as the metadata is cryptographically signed by the respective endpoint (OASIS SAML v2.0 standard, [12]).

Entities may optionally publish their metadata document locations in a zone of their corresponding DNS (RFC 1034, [13]). It is recommended, but not required, to publish the resource record in signed zone files using DNSSEC (RFC 2535, [14]).

Since this feature is an optional part of the SAML standard and requires implementing several additional standards such as Name Authority Pointer (NAPTR) Resource Records (RFC 2915, [15] and RFC 3403 [16]), no major cloud platform providers nor SAML-based Identity and Access Management (IAM) systems support it at this writing.

7.3 Pointers in signed documents

Digitally signed documents point to a natural person, be it a private individual or a person acting on behalf of an organisation. Pointers to other resources can be simply embedded to the document itself in both human-readable and human-verifiable form.

As anything outside the signed document itself can be deleted, moved or modified on purpose or inadvertently, any external resources should be embedded within the signed document itself.

8. Complementary mechanisms for the discovery

Nowadays, as most of the software clients are web-based and mobile, it is worthy to mention the standard OpenID Connect [2]. It is a simple identity layer on top of protocol OAuth 2.0, that is commonly used for user authentication performing a delegation of this process to an identity provider.

The following section is a proposal to use the OpenID Connect protocol as a complementary customized mechanism to discover the trust schemes. OpenID Connect is becoming a common authentication protocol for mobile devices and it is not difficult to find an app that prompts the user to authenticate using any social network using OpenID Connect.

In contrast to OAuth 2.0, OpenID Connect does not define standard methods to provide identity information. In order to provide information about the OpenID Providers, OpenID Connect protocol suite includes several extensions that can be used, among others, to automatic discovery of a user's OpenID Provider.

If OpenID Connect is used during the user authentication, the discovery extension of the OpenID Connect (OpenID Connect Discovery 1.0 [17]) could be used in determining the user's preferred identity provider and its configuration. This mentioned identity provider can be customized to discover the trust schemes instead, making the TSPA as the identity provider and the service provider as the ATV.

The protocol used for discovery is known as WebFinger [18]. This discovery process is based on usernames that reference hosts. By querying such a specified host, the ATV can discover the URL of the proper TSPA. The ATV sends a WebFinger request to the host extracted from the issuer of the electronic transaction in order to determine the user's preferred TSPA. Based on this information, the WebFinger service responds with a URL containing the TSPA.

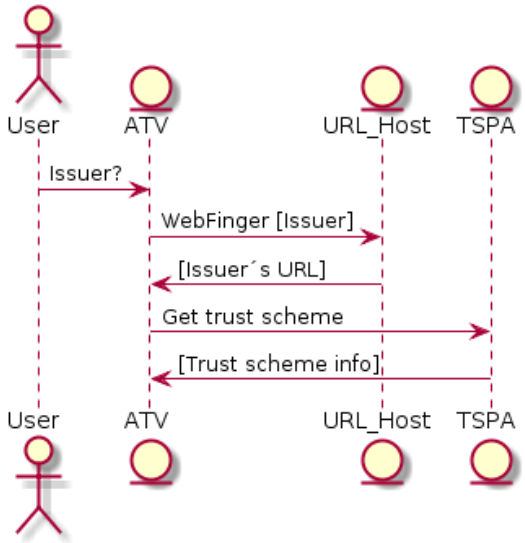

Figure 3: Example of using Webfinger in LIGHTest

In the following lines, an example of the WebFinger can be read.

The ATV would perform a WebFinger query looking for the OpenID Connect provider, in our case the TSPA. Since ATV is only looking for one particular link relation, the WebFinger resource will use the *rel* parameter [18].

```
GET /.well-known/webfinger?
      resource=eSeal.example.com&
      rel=http%3A%2F%example%2F1.0%2Fissuer
      HTTP/1.1
Host: example.com
```

The server would respond in this way:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: application/jrd+json
{
  "subject" : "eSeal.example.com",       "links" :
  [
    {
      "rel" : "http://example/issuer",
      "href" : _scheme._trust .eSeal.example.com"
    }
  ]
}
```

9. Demonstration for Selected Trust Schemes

To demonstrate the consolidated approach for the discovery process of Trust Schemes Memberships (see Section 5 and Section 6) this chapter provides examples of usage. Hereby, the examples for eIDAS Germany and the Electronic Signature Law of the People's Republic of China from the D3.3 "DNS-based Publication of Trust Schemes" are revisited and extended, in particular demonstrating the authenticity of trust declarations.

9.1 eIDAS Germany

This example shows the trust verification for a certificate issued by an eIDAS qualified trust service [19]. For clarity, real domain names of the players involved but in highly speculative ways are used.

It is assumed that the electronic transaction is simply a signed document. It is signed with a certificate issued by the German D-Trust GmbH, which is a qualified trust service provider for Germany. Either the certificate contains the Issuer Alternative Name extension with a domain name value of *d-trust.net* or the issuer certificate used for signing the certificate contains the Subject Alternative Name with that domain name value.

In order to discover the trust scheme(s) this trust service is a member, the verifier will perform a DNS query for PTR records (Step 1 in the sequence diagram in Figure 2) at the name *_scheme._trust.d-trust.net*:

```
;; QUESTION SECTION:
;_scheme._trust.d-trust.net.   IN   PTR

;; ANSWER SECTION:

_scheme._trust.d-trust.net.    IN   PTR   _scheme._trust.nrca-ds.de.
```
[2]

This indicates to the verifier that D-Trust claims a membership with a trust scheme identified as *nrca-ds.de* (which is the domain used by the German Bundesnetzagentur for their eIDAS trusted list). The verifier will have to discover the trust list for that via another DNS query (Step 3 in the sequence diagram in Figure 2):

```
;; QUESTION SECTION:
;_scheme._trust.nrca-ds.de.    IN   URI

;; ANSWER SECTION:
```

2 Note that the entries in the answer section differ to the corresponding entries in D3.3 in Section 7. It is modified and the declaration prefix _scheme._trust. is added to the PTR record (see also Section 5.3).

_scheme._trust.nrca-ds.de. IN URI https://www.nrca-ds.de/st/TSL-XML.xml

It will now download that list and see if the issuer certificate from the electronic transaction appears on that list.

Since the list is signed, it will also check whether the certificate used for signing the trusted list is valid (Step 7 in the sequence diagram in Figure 2). Again, DNS:

```
;; QUESTION SECTION:
;_scheme._trust.nrca-ds.de.   IN  SMIMEA

;; ANSWER SECTION:

_scheme._trust.nrca-ds.de.   IN  SMIMEA  <SMIMEA record data>
```

Assuming that all of this checks out, the verifier now knows that the electronic transaction was signed by a certificate issued under the trust scheme *nrca-ds.de*

9.1.1 Certificate used for signature

For this last check, whether the certificate used for signing the trusted list is valid or not the corresponding *<SMIMEA record data>* needs to be generated from the certificate used to sign the trusted list. This information can be obtained from the signature part of the trusted list.

The URI resource record refers to this published Trust Service Status List. For this example the URI *https://www.nrca-ds.de/st/TSL-XML.xml* refers to this published Trust Service Status List of Germany (see also D3.3 [20]). Here only the part of the Trust Service Status List, which is relevant for the demonstrating the authenticity of trust declarations is presented. It is located at the end of the document (starting with <ds:Signature in the example below).

```
<?xml version="1.0" encoding="UTF-8"?>
<TrustServiceStatusList
      xmlns=http://uri.etsi.org/02231/v2#
      xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"
      xmlns:ns3="http://uri.etsi.org/02231/v2/additionaltypes#"
      xmlns:ns4="http://uri.etsi.org/01903/v1.3.2#"
      xmlns:ns5="http://uri.etsi.org/TrstSvc/SvcInfoExt/eSigDir-1999-93-EC-
      TrustedList/#"
      xmlns:ns6=http://uri.etsi.org/01903/v1.4.1# Id="TrustServiceStatusList-
      1"
      TSLTag="http://uri.etsi.org/19612/TSLTag">
      <SchemeInformation>
            <TSLVersionIdentifier>5</TSLVersionIdentifier>
            <TSLSequenceNumber>61</TSLSequenceNumber>
            <TSLType>http://uri.etsi.org/TrstSvc/TrustedList/TSLType/EUgeneric
            </TSLType>
            <SchemeOperatorName>
```

```
                    <Name xml:lang="en">Federal Network Agency</Name>
                    <Name xml:lang="de">Bundesnetzagentur</Name>
              </SchemeOperatorName>
…
…
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"Id="id-
      177987bd9b07ee0b175c455e00260704">
      <ds:SignedInfo>
            <ds:CanonicalizationMethod
            Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-
            more#rsa-sha256"/>
            <ds:Reference Id="xml_ref_id" Type="" URI="">
            <ds:Transforms>
                  <ds:Transform
                  Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
                  signature"/>
                  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
                  c14n#"/>
            </ds:Transforms>
            <ds:DigestMethod
            Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <ds:DigestValue>v3oGrIOMp0EbGmzlhcQIbsT7WplPxw5IGPs0QRTFEwE=</ds:D
            igestValue>
            </ds:Reference>
                  <ds:Reference
                  Type="http://uri.etsi.org/01903#SignedProperties"
                  URI="#xades-id-177987bd9b07ee0b175c455e00260704">
                  <ds:Transforms>
                        <ds:Transform
                        Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
                  </ds:Transforms>
                  <ds:DigestMethod
                  Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
                  <ds:DigestValue>mWNqN6WjCIib6jTGBNZ+ZxrXnzqVhTAPbF5V5Y7tpt4=
                  </ds:DigestValue>
            </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue Id="value-id-177987bd9b07ee0b175c455e00260704">
      OxuuBnGORvL+unCUIto/uA/eHqWnC7thaOR40o+B2JzRGMtS40ShLqpgtfEGMwMKRazv6sZ8
      dFTRe+rfQgWhxAsXJL/sEUSrbvPv8djOgcsV8iImL9VhUXCpM1qFziaMSH/xJxdQ4RY/IVpU
      +0rPT0BbQJ24UOXt7J9D8sdnBb3IpAtydQzSQHKLjcJYEwPI5ZW8WgtrmdtEkYHgOE6GCLgW
      Izw0nZHU6ps3OJtiDiUe6ClRLxlwD+ZdhiGgG7tn/tXBgW8sHLP3AlsmlT/bTW8fyRVzrRd/
      Jq+1e2SZIsdw5P8GBh5BxanhFg/jGeLuf+sy41brERt3UkwSJj5c3A==
      </ds:SignatureValue><
      ds:KeyInfo>
            <ds:X509Data>
            <ds:X509Certificate>MIIECjCCAvKgAwIBAgICBH8wDQYJKoZIhvcNAQENBQAwPz
            ELMAkGA1UEBhMCREUxGjAYBgNVBAoMEUJ1bmRlc25ldHphZ2VudHVyMRQwEgYDVQQD
            DAsxNFItQ0EgMTpQTjAeFw0xNDA0MTEwODQ0NTJaFw0xOTA0MTEwNjM1MDBaMEAxCz
            AJBgNVBAYTAkRFMRowGAYDVQQKDBFCdW5kZXNuZXR6YWdlbnR1cjEVMBMGA1UEAwwM
            MTRSLVRTTCAxOlBOMIIBIzANBgkqhkiG9w0BAQEFAAOCARAAMIIBCwKCAQEAkyyMPd
            tWEDtPcT+eq+KKYaQ5G+6Hbpl9i6b3nBN6+3DROzqaVqtehrCpuE5CmUdqR2lixvHT
```

```
bEjYIlk3jsmPTxtImfZ66mwKUoenulI6jE5/lvRNtqKWQbLTd7nrEJAecy/ouHWZ6x
iDB/ytftxJhAREUqGPfJiWnCFoyRrDSW6GQ8QQbJnlHMLuxs30KNUIRbVOOX/jb8oe
qFI0zXUeSH/AMrshRM3G8W941tee8nn5jK2CZvjOuYEI1hNpcXAzBTuaFRJhLdsvg0
SfgW0T6tFhuUbG5eW9wraGOMCNdzfcNnjmFitVrBRtl9yIfyVn2Tgd2DfJ9cHLJGmb
TBnUIwIEQAAAgaOCAQwwggEIMA4GA1UdDwEB/wQEAwIGQDAdBgNVHQ4EFgQUYqVd8y
HV7CHE+JCq3zLhvyLM43wwEQYDVR0lBAowCAYGBACRNwMAMBgGCCsGAQUFBwEDBAww
CjAIBgYEAI5GAQEwHwYDVR0jBBgwFoAU/fNQhDCO7COa9TOy44EH3eTvgK4wSgYIKw
YBBQUHAQEEPjA8MDoGCCsGAQUFBzABhi5odHRwOi8vb2NzcC5ucmNhLWRzLmRlOjgw
ODAvb2NzcC1vY3NwcmVzcG9uZGVyMBIGA1UdIAQLMAkwBwYFKyQIAQEwGwYJKwYBBA
HAbQMFBA4wDAYKKwYBBAHAbQMFATAMBgNVHRMBAf8EAjAAMA0GCSqGSIb3DQEBDQUA
A4IBAQAMm2Fj5hoZBOeOOT4LPrky39cTYMPN1+Patx6BB+kuF/pXAI/GmDyOuFIZ+/
Sf8bz336sbbIfnbDeV6Y6ZJvCnqzrUT8kBlf3+QTQ+JxOEYfw1bdRffjmYDCbM0S7R
w02eAaSykiHSkSp8kWA6rYWkhVakX/v/PdBUtkPHdq1P5ghLPx7Gk/ax/U3fDLlKGm
s5iJjz55AIMqlK4HWEc7xZk3QoD8w+lpRqT5QNYwex5ueXO/Mpd9ZtY5qm7bJKhRnK
ejQaaMO1frAWT+QM2Uve3TaZlgupa0K+FL9i532dMd/D4RjxtDTNfa5o8gcNFS6eDy
uo0z8BJDp9LCLtNZYT
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
<ds:Object>
<xades:QualifyingProperties
xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"
Target="#id-177987bd9b07ee0b175c455e00260704">
<xades:SignedProperties Id="xades-id-
177987bd9b07ee0b175c455e00260704">
<xades:SignedSignatureProperties>
<xades:SigningTime>2017-12-01T13:12:59Z</xades:SigningTime>
<xades:SigningCertificate>
<xades:Cert>
<xades:CertDigest>
<ds:DigestMethod
Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<ds:DigestValue>4dsvmQtp3SV5/ysk+dpRX2eP4QQ=<
/ds:DigestValue>
</xades:CertDigest>
<xades:IssuerSerial>
<ds:X509IssuerName>CN=14R-
CA1:PN,O=Bundesnetzagentur,C=DE</ds:X509IssuerName>
<ds:X509SerialNumber>1151</ds:X509SerialNumber>
</xades:IssuerSerial>
</xades:Cert>
</xades:SigningCertificate>
</xades:SignedSignatureProperties>
<xades:SignedDataObjectProperties>
<xades:DataObjectFormat ObjectReference="#xml_ref_id">
<xades:MimeType>application/octet-
stream</xades:MimeType>
</xades:DataObjectFormat>
</xades:SignedDataObjectProperties>
</xades:SignedProperties>
```

```
            </xades:QualifyingProperties>
      </ds:Object>
      </ds:Signature>
</TrustServiceStatusList>
```

The certificate used to sign the Trust Service Status List, which shall be verified, is given in the `<ds:X509Certificate>` element as a Base64 encoded string of the DER encoded X.509 certificate. The following figure shows some relevant information in its decoded form:

```
Version: 3 (0x2)
Serial Number: 1151 (0x47f)
Issuer: C=DE, O=Bundesnetzagentur, CN=14R-CA 1:PN
Validity
     Not Before: Apr 11 08:44:52 2014 GMT
     Not After : Apr 11 06:35:00 2019 GMT
Subject: C=DE, O=Bundesnetzagentur, CN=14R-TSL 1:PN
Subject Public Key Info:
     Public Key Algorithm: rsaEncryption
     Public-Key: (2048 bit)
     Modulus:
          00:93:2c:8c:3d:db:56:10:3b:4f:71:3f:9e:ab:e2:
          8a:61:a4:39:1b:ee:87:6e:99:7d:8b:a6:f7:9c:13:
          7a:fb:70:d1:3b:3a:9a:56:ab:5e:86:b0:a9:b8:4e:
          42:99:47:6a:47:69:62:c6:f1:d3:6c:48:d8:22:59:
          37:8e:c9:8f:4f:1b:48:99:f6:7a:ea:6c:0a:52:87:
          a7:ba:52:3a:8c:4e:7f:96:f4:4d:b6:a2:96:41:b2:
          d3:77:b9:eb:10:90:1e:73:2f:e8:b8:75:99:eb:18:
          83:07:fc:ad:7e:dc:49:84:04:44:52:a1:8f:7c:98:
          96:9c:21:68:c9:1a:c3:49:6e:86:43:c4:10:6c:99:
          e5:1c:c2:ee:c6:cd:f4:28:d5:08:45:b5:4e:39:7f:
          e3:6f:ca:1e:a8:52:34:cd:75:1e:48:7f:c0:32:bb:
          21:44:cd:c6:f1:6f:78:d6:d7:9e:f2:79:f9:8c:ad:
          82:66:f8:ce:b9:81:08:d6:13:69:71:70:33:05:3b:
          9a:15:12:61:2d:db:2f:83:44:9f:81:6d:13:ea:d1:
          61:b9:46:c6:e5:e5:bd:c2:b6:86:38:c0:8d:77:37:
          dc:36:78:e6:16:2b:55:ac:14:6d:97:dc:88:7f:25:
          67:d9:38:1d:d8:37:c9:f5:c1:cb:24:69:9b:4c:19:
          d4:23
     Exponent: 1073741953 (0x40000081)
X509v3 extensions:
     X509v3 Subject Key Identifier:
          62:A5:5D:F3:21:D5:EC:21:C4:F8:90:AA:DF:32:E1:BF:22:CC:E3:7C
     X509v3 Authority Key Identifier:
          FD:F3:50:84:30:8E:EC:23:9A:F5:33:B2:E3:81:07:DD:E4:EF:80:AE
     Authority Information Access:
          OCSP - URI:http://ocsp.nrca-ds.de:8080/ocsp-ocspresponder
```

Note that the Trust Service Status List only contains this one certificate. It does not contain or provides a way to acquire the issuer certificate used to sign this certificate beyond mentioning its subject name in the 'Issuer' field and its key identifier in the 'Authority Key Identifier' extension.

This means that without further knowledge, classical PKIX (Public Key Infrastructure for X.509 Certificates: http://www.itu.int/rec/T-REC-X.509/en) verification of the certificate by building a chain to a set of trusted root certificates is not possible.

9.1.2 Certificate constraints using SMIMEA

With the SMIMEA resource record, the certificates that are accepted for the signature of the Trust Service Status List can be limited. As already mentioned in the sections 5.4 and 6.1, the format of the SMIMEA resource record is the same as for the TLSA record and the different possibilities to limit the certificates are defined in RFC6698 [9]. In addition, RFC7218 [21] defines acronyms for the three numeric fields for simplification. A short description of the SMIMEA/TLSA fields is provided in the following. Further details of the SMIMEA/TLSA fields can be found in the corresponding RFCs [9] and [21], as well as in D2.7 [7].

The Certificate Usage field describes how the certificate or public key data provided in the record should be used for validation. Table 1 lists the four currently defined constraints.

Table 1: TLSA Certificate Usages; adapted from RFC7218 [21]

Value	Acronym	Short Description
0	PKIX-TA	CA constraint
1	PKIX-EE	Service certificate constraint
2	DANE-TA	Trust anchor assertion
3	DANE-EE	Domain-issued certificate

As the acronyms are trying to call out by consisting of two parts, these constraints are a combination of two underlying features.

The first of these is whether traditional PKIX validation should be used – that is, whether a validator should attempt to build a validation chain starting from its own set of trusted root certificates –, or whether validation should solely be based on information provided via the record.

Secondly, the record can either contain a certificate or public key which should be used in building a validation chain. i.e., it contains a trust anchor or TA for validation, or it can contain the specific certificate or key to be used when signing. Such a certificate is known as the 'end entity' or EE certificate.

These two parts combine to the four constraints:

- The *CA Constraint* specifies a CA certificate or public key which must be part of the validation chain for the certificate used for signing when constructing such a chain via traditional PKIX procedures.
- The *Service Certificate Constraint* specifies the certificate or public key that must match the certificate used for signing. Additionally, the certificate needs to be validated using PKIX procedures.
- The *Trust Anchor Assertion* specifies a certificate or public key that must be used as the starting point for building a validation chain for the certificate used for singing.
- The *Domain-issued Certificate* specifies the certificate or public key that must match the certificate used for signing. No additional validation is required.

The first two constraints rely on an agreed upon set of trusted root certificates for validation. Such a set has been established by the Mozilla CA Certificate Store (https://www.mozilla.org/en-US/about/governance/policies/security-group/certs/). It is, however, specifically intended for encrypted connections to web servers only and therefore is not necessarily for certificates used in signing document. Instead of either relying on this list anyway or attempting to establish an alternative set, it is more robust to only use the two latter constraints that do not rely on such a set.

Which of those two constraints should be used depends mostly on policy. When using *DANE-TA*, the certificate of the CA issuing the certificates used for signing the Trust Service Status List is published. This makes it possible to change the actual certificate used, for instance in case it was compromised, without also having to update the DNS records. Conversely, if *DANE-EE* is used the records need to be updated. But this constraint means that a client validating the list doesn't need to consider any additional certificates.

With the Selector field it is specified which part of the certificate will be matched against the associated data, either the full certificate or the certificate's public key. The corresponding values and acronyms for the Selector field are listed in Table 2.

Table 2: TLSA Selectors; adapted from RFC7218 [21]

Value	Acronym	Short Description
0	Cert	Full certificate
1	SPKII	SubjectPublicKeyInfo

Here, the choice is between limiting to either a specific certificate or all certificates that contain a public key. Choosing the later would allow to issue a new certificate for the same key, for instance because the old certificate expired, without the need to update the SMIMEA records.

Because the full certificate is available as part of Trust Service Status List, referring to it instead of just part of it probably makes handling easier and might be preferred.

The Matching Type field determines the format of the certificate association data. It can be specified if an exact match, or a SHA-256/512 hash of the selected content should be used. The corresponding values and acronyms for the Matching Type field are listed inTable 3.

Table 3: TLSA Matching Types; adapted from RFC7218 [21]

Value	Acronym	Short Description
0	Full	No hash used
1	SHA2-256	256 bit hash by SHA2
2	SHA2-512	512 bit hash by SHA2

Whether to choose an exact match or one of the hashes depends mainly on whether the certificate in question is already available at the verifier. Only in Matching Type 0 is the certificate or public key necessary for checking signatures present in the record and can be used directly. If either of the hashes is chosen, the data in the record can only be used to verify that the correct certificate or public key has been presented via other means.

Because certificate usage *DANE_EE* refers to the actual certificate used for signing and this certificate is provided as part of the Trust Service Status List itself, there is no need to include it in full in the SMIMEA record. Since the hashes are significantly shorter, and at this time, SHA-256 provides reasonable security, this means the records will be significantly shorter.

9.1.3 SMIMEA records for eIDAS Germany

For the eIDAS Germany example, the two possible SMIMEA resource records would look like this based on the recommendations from above:

```
_scheme._trust.nrca-ds.de.   IN  SMIMEA  (
            3                       ; certificate usage domain issued cert
            0                       ; selector: full certificate
            1                       ; matching type SHA-256
            c70cd54924d4c9cf        ; certificate association data
            6ed20dc93c76aabb
            45ae2af3b5cf3de9
            e4fb1224b6b58646
)
```

and:

```
_scheme._trust.nrca-ds.de.   IN  SMIMEA  (
            2                       ; certificate usage trust anchor assertion
```

```
                0                       ; selector: full certificate
                0                       ; matching type full certificate
                3082040a308202f2        ; certificate association data
                a0030201020202047f300d06092a864886f70d01010d0500303f310b3009060355
                040613024445311a3018060355040a0c1142756e6465736e65747a6167656e7475
                72311430120603550403 0c0b3134522d434120313a504e301e170d313430343131
                3038343435325a170d3139303431313036333530305a3040310b30090603550406
                13024445311a3018060355040a0c1142756e6465736e65747a6167656e74757231
                15301306035504030c0c3134522d54534c20313a504e30820123300d06092a8648
                86f70d01010105000382011000308201 0b0282010100932c8c3ddb56103b4f713f
                9eabe28a61a4391bee876e997d8ba6f79c137afb70d13b3a9a56ab5e86b0a9b84e
                4299476a476962c6f1d36c48d82259378ec98f4f1b4899f67aea6c0a5287a7ba52
                3a8c4e7f96f44db6a29641b2d377b9eb10901e732fe8b87599eb188307fcad7edc
                4984044452a18f7c98969c2168c91ac3496e8643c4106c99e51cc2eec6cdf428d5
                0845b54e397fe36fca1ea85234cd751e487fc032bb2144cdc6f16f78d6d79ef279
                f98cad8266f8ceb98108d61369717033053b9a1512612ddb2f83449f816d13ead1
                61b946c6e5e5bdc2b68638c08d7737dc3678e6162b55ac146d97dc887f2567d938
                1dd837c9f5c1cb24699b4c19d423020440000081a382010c30820108300e060355
                1d0f0101ff040403020640301d0603551d0e0416041462a55df321d5ec21c4f890
                aadf32e1bf22cce37c30110603551d25040a300806060400913703003018060 82b
                0601050507010 3040c300a3008060604008e460101301f0603551d230418301680
                14fdf35084308eec239af533b2e38107dde4ef80ae304a06082b06010505070101
                043e303c303a06082b06010505073001862e687474703a2f2f6f6373702e6e7263
                612d64732e64653a383038302f6f6373702d6f637370726573706f6e6465723012
                0603551d20040b3009300706052b24080101301b06092b06010401c06d0305040e
                300c060a2b06010401c06d030501300c0603551d130101ff04023000300d06092a
                864886f70d01010d050003820101000c9b6163e61a1904e78e393e0b3eb932dfd7
                1360c3cdd7e3dab71e8107e92e17fa57008fc6983c8eb85219fbf49ff1bcf7dfab
                1b6c87e76c3795e98e9926f0a7ab3ad44fc90195fdfe41343e27138461fc356dd4
                5f7e39980c26ccd12ed1c34d9e01a4b29221d2912a7c91603aad85a48556a45ffb
                ff3dd054b643c776ad4fe6084b3f1ec693f6b1fd4ddf0cb94a1a6b398898f3e790
                0832a94ae0758473bc599374280fcc3e96946a4f940d6307b1e6e7973bf32977d6
                6d639aa6edb24a8519ca7a341a68c3b57eb0164fe40cd94bdedd3699960ba96b42
                be14bf62e77d9d31dfc3e118f1b434cd7dae68f2070d152e9e0f2ba8d33f01243a
                7d2c22ed359613
)
```

9.2 Electronic Signature Law of the People's Republic of China

The People's Republic of China do not have a public trust list published under any domain presently. Therefore, the process described here is from an educated guess on how it should be accessed if it was published on a domain. We use the eIDAS format as a template for China also.

It is assumed that the electronic transaction is simply a signed document. It is signed with a certificate issued by the Chinese Trust Provider (A fictional trust provider), which is a qualified trust service for China. Either the certificate contains the Issuer Alternative Name extension with a domain name value of *ctrustprov.cn* or the issuer certificate used for signing the certificate contains the Subject Alternative Name with that domain name value.

In order to discover the trust scheme(s) this trust service is a member, the verifier will perform a DNS query for PTR records (Step 1 in the sequence diagram in Figure 2) at the name *_scheme._trust.ctrustprov.cn*:

```
;; QUESTION SECTION:
;_scheme._trust.ctrustprov.cn.example.  IN  PTR

;; ANSWER SECTION:

_scheme._trust.ctrustrov.cn.example.   IN  PTR

_scheme._trust.mict.gov-cn.example.
```

This indicates to the verifier that Chinese Trust claims a membership with a trust scheme identified as *mict.gov-cn.example* (which is the fictional domain used by the Chinese for their trusted list owned by Ministry of Information and Communication Technology in China). The verifier will have to discover the trust list for that via another DNS query (Step 3 in the sequence diagram in Figure 2):

```
;; QUESTION SECTION:
;_scheme._trust.mict.gov-cn.example.   IN  URI

;; ANSWER SECTION:
_scheme._trust.mict.gov-cn.example.   IN  URI  https://www.
mict.gov-cn.example./ct/TSL-XML.xml
```

It will now download that list and see if the issuer certificate from the electronic transaction appears on that list.

Since the list is signed, it will also check whether the certificate used for signing the trusted list is valid (Step 7 in the sequence diagram in Figure 2). Again, DNS:

```
;; QUESTION SECTION:
;_scheme._trust.mict.gov-cn.example.      IN  SMIMEA
```

```
;; ANSWER SECTION:

;_scheme._trust.mict.gov-cn.example.   IN  SMIMEA  <SMIMEA record data>
```

Assuming, that all of this checks out, the verifier now knows that the electronic transaction was signed by a certificate issued through the trust scheme *mict.gov-cn.example.*

The URI resource record refers to a fictiticious Trust Service Status List of the record of China. Here only the part of the Trust Service Status List, which is relevant for the demonstrating the authenticity of trust declarations is presented.

```
<?xml version="1.0" encoding="UTF-8"?>
<TrustServiceStatusList
      xmlns="http://uri.etsi.org/02231/v2#"
      xmlns:ns2="http://www.w3.org/2000/09/xmldsig#"
      xmlns:ns3="http://uri.etsi.org/02231/v2/additionaltypes#"
      xmlns:ns4="http://uri.etsi.org/01903/v1.3.2#"
      xmlns:ns5="http://uri.etsi.org/TrstSvc/SvcInfoExt/eSigDir-1999-93-EC-
      TrustedList/#"
      xmlns:ns6="http://uri.etsi.org/01903/v1.4.1#"
      Id="TrustServiceStatusList-1"
      TSLTag="http://uri.etsi.org/19612/TSLTag">
      <SchemeInformation>
            <TSLVersionIdentifier>4</TSLVersionIdentifier>
            <TSLSequenceNumber>23</TSLSequenceNumber>
            <TSLType>http://uri.etsi.org/TrstSvc/TrustedList/TSLType/CNList</T
            SLType>
            <SchemeOperatorName>
            <Name xml:lang="en"> Ministry of Information and Communication
            Technology</Name>
            </SchemeOperatorName>
      …
      …
      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"Id="id-
      177987bd9b07ee0b175c455e00260704">
      <ds:SignedInfo>
            <ds:CanonicalizationMethod
            Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            <ds:SignatureMethod Algorithm="
            http://www.w3.org/2009/xmldsig11#ECKeyValue"/>
            <ds:Reference Type="http://uri.etsi.org/01903#SignedProperties"
            URI="#xades-id-177987bd9b07ee0b175c455e00260704">
            <ds:Transforms>
                  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
                  c14n#"/>
            </ds:Transforms>
            <ds:DigestMethod
            Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
            <ds:DigestValue>AKerEbWIzkOwfB/yTxXDBasaLwTYQiT1vr0sqKt9tCA=</ds:D
            igestValue>
            </ds:Reference>
```

```
</ds:SignedInfo>
<ds:SignatureValue Id="value-id-177987bd9b07ee0b175c455e00260704">
MHcCAQEEIClI32JRY6QppxZQNlsGJBdJHf/T2mazMjHOE8Lm6XGXoAoGCCqGSM49
AwEHoUQDQgAE0CugLAXhigr2k4VADyqvkrrK809wBpgPmTwWaWiq3YY0xvca++76
XBFkGadco5wSbJWqZhFxnF4ke4dPfOmU7g==
</ds:SignatureValue><
ds:KeyInfo>
      <ds:X509Data>
      <ds:X509Certificate>
      MIICmTCCAkACCQDlUGdWSJob5TAKBggqhkjOPQQDAjCB1DELMAkGA1UEBhMCQ04x
      EDAOBgNVBAgMB0JlaWppbmcxEDAOBgNVBAcMB0JlaWppbmcxODA2BgNVBAoML01p
      bmlzdHJ5IG9mIEluZHVzdHJ5IGFuZCBJbmZvcm1hdGlvbiBUZWNobm9sb2d5MR8w
      HQYDVQQLDBZJbmZvcm1hdGlvbiBUZWNobm9sb2d5MRwwGgYDVQQDDBNtaWN0Lmdv
      di1jbi5leGFtcGxlMSgwJgYJKoZIhvcNAQkBFhlhZG1pbkBtaWN0Lmdvdi1jbi5l
      eGFtcGxlMB4XDTE4MDUwMzEzNDUwOFoXDTE5MDkxNTEzNDUwOFowgdQxCzAJBgNV
      BAYTAkNOMRAwDgYDVQQIDAdCZWlqaW5nMRAwDgYDVQQHDAdCZWlqaW5nMTgwNgYD
      VQQKDC9NaW5pc3RyeSBvZiBJbmR1c3RyeSBhbmQgSW5mb3JtYXRpb24gVGVjaG5v
      bG9neTEfMB0GA1UECwwWSW5mb3JtYXRpb24gVGVjaG5vbG9neTEcMBoGA1UEAwwT
      bWljdC5nb3YtY24uZXhhbXBsZTEoMCYGCSqGSIb3DQEJARYZYWRtaW5AbWljdC5n
      b3YtY24uZXhhbXBsZTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABNAroCwF4YoK
      9pOFQA8qr5K6yvNPcAaYD5k8Fmloqt2GNMb3Gvvu+lwRZBmnXKOcEmyVqmYRcZxe
      JHuHT3zplO4wCgYIKoZIzj0EAwIDRwAwRAIgBwcrqRBzVnt5ZuQJ5dztR/xD+aw2
      JQ8j4+9BYiljQ9gCICYx6CjMbrGjrmFM/YxTvYT9AqRvtEaaccDGKP3YOuHU
      </ds:X509Certificate>
      </ds:X509Data>
</ds:KeyInfo>
<ds:Object>
      <xades:QualifyingProperties
      xmlns:xades="http://uri.etsi.org/01903/v1.3.2#"
      Target="#id-177987bd9b07ee0b175c455e00260704">
      <xades:SignedProperties Id="xades-id-
      177987bd9b07ee0b175c455e00260704">
      <xades:SignedSignatureProperties>
      <xades:SigningTime>2017-12-01T13:12:59Z</xades:SigningTime>
      <xades:SigningCertificate>
      <xades:Cert>
            <xades:CertDigest>
                  <ds:DigestMethod
                  Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                  <ds:DigestValue>
                  mo740Os0z10bDbCRq2km4TqiSjQP6LgSDezZfWxcu+g=
                  </ds:DigestValue>
            </xades:CertDigest>
            <xades:IssuerSerial>
                  <ds:X509IssuerName> E = admin@mict.gov-cn.example,CN =
                  mict.gov-cn.example,OU = Information Technology,O =
                  Ministry of Information and Communication Technology,C
                  = CN</ds:X509IssuerName>
                  <ds:X509SerialNumber>
                  00e5506756489a1be5</ds:X509SerialNumber>
```

```
                  </xades:IssuerSerial>
            </xades:Cert>
            </xades:SigningCertificate>
            </xades:SignedSignatureProperties>
            <xades:SignedDataObjectProperties>
                  <xades:DataObjectFormat ObjectReference="#xml_ref_id">
                        <xades:MimeType>application/octet-
                        stream</xades:MimeType>
                  </xades:DataObjectFormat>
            </xades:SignedDataObjectProperties>
            </xades:SignedProperties>
            </xades:QualifyingProperties>
      </ds:Object>
      </ds:Signature>
</TrustServiceStatusList>
```

The certificate used to sign the Trust Service Status List is given in the `<ds:X509Certificate>` element as a Base64 encoded string of the DER encoded X.509 certificate. The following figure shows some relevant information in its decoded form:

```
Identity:               mict.gov-cn.example
Verified by:            mict.gov-cn.example
Expires:                09/15/2019

Subject Name
C (Country):            CN
ST (State):             Beijing
L (Locality):           Beijing
O (Organization):       Ministry of Information and Communication Technology
OU (Organizational Unit):   Information Technology
CN (Common Name):       mict.gov-cn.example
EMAIL (Email Address):  admin@mict.gov-cn.example

Issuer Name
C (Country):            CN
ST (State):             Beijing
L (Locality):           Beijing
O (Organization):       Ministry of Information and Communication Technology
OU (Organizational Unit):   Information Technology
CN (Common Name):       mict.gov-cn.example
EMAIL (Email Address):  admin@mict.gov-cn.example

Issued Certificate
Version:                1
Serial Number:          00 E5 50 67 56 48 9A 1B E5
Not Valid Before:       2018-05-03
Not Valid After:        2019-09-15
Certificate Fingerprints
```

```
SHA1:                   C1 12 42 2A 8B AF 59 1A 32 44 FC CF 4B 47 99 A2 B6 5B
96 9F
MD5:                    B5 79 D3 90 E7 91 A9 8D 16 29 48 1A AA E9 19 F5
Public Key Info
Key Algorithm:          Elliptic Curve
Key Parameters:         06 08 2A 86 48 CE 3D 03 01 07
Key Size:               256
Key SHA1 Fingerprint:   2B C6 86 2D 5E C7 20 3A 7B AC EF 10 65 E9 FE B4 67 DF
43 2D
Public Key:             04 D0 2B A0 2C 05 E1 8A 0A F6 93 85 40 0F 2A AF 92 BA
                        CA F3 4F 70 06 98 0F 99 3C 16 69 68 AA DD 86 34 C6 F7
                        1A FB EE FA 5C 11 64 19 A7 5C A3 9C 12 6C 95 AA 66 11
                        71 9C 5E 24 7B 87 4F 7C E9 94 EE

Signature
Signature Algorithm:    SHA256 with ECDSA
Signature:              30 44 02 20 07 07 2B A9 10 73 56 7B 79 66 E4 09 E5 DC
                        ED 47 FC 43 F9 AC 36 25 0F 23 E3 EF 41 62 29 63 43 D8
                        02 20 26 31 E8 28 CC 6E B1 A3 AE 61 4C FD 8C 53 BD 84
                        FD 02 A4 6F B4 46 9A 71 C0 C6 28 FD D8 3A E1 D4
```

However, there is a root certificate used to sign this certificate. This certificate has a longer lifetime that the one used to sign the trust list because it is not used to sign every trust list generated. The details are below

```
mict.gov-cn.example
Identity:               mict.gov-cn.example
Verified by:            mict.gov-cn.example
Expires:                02/20/2021

Subject Name
C (Country):            CN
ST (State):             Beijing
L (Locality):           Beijing
O (Organization):       Ministry of Information and Communication Technology
OU (Organizational Unit):   Information Technology
CN (Common Name):       mict.gov-cn.example
EMAIL (Email Address):  admin@mict.gov-cn.example

Issuer Name
C (Country):            CN
ST (State):             Beijing
L (Locality):           Beijing
O (Organization):       Ministry of Information and Communication Technology
OU (Organizational Unit):   Information Technology
CN (Common Name):       mict.gov-cn.example
EMAIL (Email Address):  admin@mict.gov-cn.example

Issued Certificate
```

```
Version: 3
Serial Number:          00 92 EF D0 E3 85 06 7C 4B
Not Valid Before:       2018-05-03
Not Valid After:        2021-02-20
Certificate Fingerprints
SHA1:                   50 46 DB 43 EB 0A 55 AC AE C8 BA 72 7F BA 20 95 2C E5
94 E6
MD5:                    D4 BE D0 38 D4 53 43 88 91 53 F0 B2 0A C7 F4 52
Public Key Info
Key Algorithm:          Elliptic Curve
Key Parameters:         06 08 2A 86 48 CE 3D 03 01 07
Key Size:               256
Key SHA1 Fingerprint:   7B 0A 98 88 71 5E DF 97 28 36 14 AB D2 F6 19 7E 85 6D
78 8F
Public Key:             04 C9 7C 10 A8 64 B9 FF 81 1C C4 62 66 47 B7 94 75 F7
                        6E 42 94 79 2A 62 89 10 AA F1 AC 99 E3 06 0E DC D2 DC
                        3F D5 62 35 73 FC A3 50 A1 17 89 32 10 99 2A 52 95 6E
                        43 DE 91 EA 51 44 05 93 95 B3 98
Subject Key Identifier
Key Identifier:         77 B7 89 80 E6 8B 36 DF C7 68 7F 94 95 03 E4 30 66 84
BE 90
Critical:               No
Extension
Identifier:             2.5.29.35
Value:                  30 16 80 14 77 B7 89 80 E6 8B 36 DF C7 68 7F 94 95 03
E4 30 66 84 BE 90
Critical:               No
Basic Constraints
Certificate Authority:  Yes
Max Path Length:        Unlimited
Critical:               No
Signature
Signature Algorithm:        SHA256 with ECDSA
Signature:                  30 44 02 20 0D 76 84 F5 47 2C D4 B1 FF 9A B3 5F 0A
A1 34 82 FB BB 4B AF 7D C0 35 B1 C3 8B F0 64 0E 21 FE DB 02 20 19 DE C2 EA 2F
F5 47 A7 83 5C 22 3A DF 66 30 B8 10 90 44 34 AB 96 99 05 BB 21 53 BD 67 12 DC
0F
```

Note that the Trust Service Status List only contains this one certificate. It does not contain or provides a way to acquire the issuer certificate used to sign this certificate beyond mentioning its subject name in the ‘Issuer’ field and its key identifier in the ‘Authority Key Identifier’ extension.

This means that without further knowledge, classical PKIX verification of the certificate by building a chain to a set of trusted root certificates is not possible.

The SMIMEA resource record allows to limit the certificates that are accepted for the signature of the Trust Service Status List. The mechanism and possible choices have been discussed

already in Section9.1.2. Using a hash value of certificate used for signing, the record for Electronic Signature Law of the People's Republic of China example could look like this:

```
_ scheme._trust.mict.gov-cn.example.   IN  SMIMEA  (
            3                       ; certificate usage domain issued cert
            0                       ; selector: full certificate
            1                       ; matching type SHA-256
            00e5506756489a1be5      ; certificate association data

)
```

10. Summary and Conclusion

The deliverable D3.4 outlines the discovery mechanisms for Trust Scheme Publication Authorities within LIGHTest. It builds on the results of the deliverables D3.1 “Conceptual Framework for Trust Schemes” and D3.3 “DNS-based Publication of Trust Schemes”.

In this deliverable, the consolidated approach to publishing trust-related information in the DNS is specified for the discovery mechanisms for Trust Scheme Publication Authorities, which includes the following: discovery of the trust scheme, verification process of the Trust Scheme Membership, verification of the authenticity of the trust list. In addition, a complementary mechanism for the discovery mechanism is outlined using the OpenID Connect protocol.

Furthermore, a concept for the discovery of the policy and rules of the trust scheme as well as a summary of direct pointers to the relevant DNS data embedded e.g. in X509.signatures is presented.

The discovery of Trust Scheme Publication Authorities is illustrated by revisiting and extending the examples for eIDAS Germany and the Electronic Signature Law of the People’s Republic of China of D3.3. With these examples, the complete chain for the association of an Issuer with a Trust Scheme including the discovery and verification of the Trust Scheme Membership as well as the verification of the authenticity of the trust list is demonstrated.

11. References

[1] ETSI TS 119 612, „Electronic Signatures and Infrastructures (ESI);Trusted Lists,“ European Telecommunications Standards Institute; Technical Specification, Sophia Antipolis Cedex, V2.1.1 (2015-07), 2015.

[2] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C., OpenID Connect Core 1.0, November 8, 2014.

[3] P. Falstrom, O. Kolkman, The Uniform Resource Identifier (URI) DNS Resource, RFC 7553, Internet Engineering Task Force, June 2015.

[4] Mockapetris, P.V., Domain names – implementation and specification, RFC 1035, Internet Engineering Task Force, November 1987.

[5] The LIGHTest Project, D3.1 - Conceptual Framework for Trust Schemes (1), Project Deliverable, 2017.

[6] The LIGHTest Project, D3.3 - DNS-based Publication of Trust Schemes, Project Deliverable, 2018.

[7] The LIGHTest Project, D2.7 -Relevant DNSSEC Concepts and Basic Building Blocks, Project Deliverable, 2017.

[8] P. Hoffmann, J. Schlyter, Using Secure DNS to Associate Certificates with Domain Names for S/MIME, RFC8162, Internet Engineering Task Force, May 2017.

[9] P. Hoffmann, J. Schlyter, The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA, RFC 6698, Internet Engineering Task Force, August 2012.

[10] The LIGHTest Project, D2.14 - Reference Architecture, Project Delivarable, 2017.

[11] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 5280, Internet Engineering Task Force, May 2008.

[12] S. Cantor, J. Moreh, R. Philpott, E. Maler, Metadata for the OASIS Security Assertion Markup Language, OASIS Standard saml-metadata-2.0-os, March 2005.

[13] P. Mockapetris, Domain Names - Concepts and Facilities, RFC 1034, Internet Engineering Task Force, November 1987.

[14] D. Eastlake, Domain Name System Security Extensions, RFC 2535, Internet Engineering Task Force, March 1999.

[15] M. Mealling, R. Daniel, The Naming Authority Pointer (NAPTR) DNS Resource Record, RFC 2915, Internet Engineering Task Force, September 2000.

[16] M. Mealling, Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database, RFC 3403, Internet Engineering Task Force, October 2002.

[17] N. Sakimura, J. Bradley, M. Jones, E. Jay,, OpenID Connect Discovery 1.0,, November 8, 2014.

[18] P. Jones, G. Salgueiro, M. Jones, J. Smarr,, WebFinger, RFC 7033 Internet Engineering Task,, September 2013.

[19] European Parliament, Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC, European Parliament, Brussels, Belgium, Regulation 910/201, 2014.

[20] The LIGHTest Project, D3.3 - DNS-based Publication of Trust Schemes, Project Deliverable, February 2018.

[21] Gudmundsson, O., Adding Acronyms to Simplify Conversations about DNS-based Authentication of Named Entities (DANE), RFC 7218, Internet Task Force, April 2014.

12. Project Description

LIGHTest project to build a global trust infrastructure that enables electronic transactions in a wide variety of applications

An ever increasing number of transactions are conducted virtually over the Internet. How can you be sure that the person making the transaction is who they say they are? The EU-funded project LIGHTest addresses this issue by creating a global trust infrastructure. It will provide a solution that allows one to distinguish legitimate identities from frauds. This is key in being able to bring an efficiency of electronic transactions to a wide application field ranging from simple verification of electronic signatures, over eProcurement, eJustice, eHealth, and law enforcement, up to the verification of trust in sensors and devices in the Internet of Things.

Traditionally, we often knew our business partners personally, which meant that impersonation and fraud were uncommon. Whether regarding the single European market place or on a Global scale, there is an increasing amount of electronic transactions that are becoming a part of peoples everyday lives, where decisions on establishing who is on the other end of the transaction is important. Clearly, it is necessary to have assistance from authorities to certify trustworthy electronic identities. This has already been done. For example, the EC and Member States have legally binding electronic signatures. But how can we query such authorities in a secure manner? With the current lack of a worldwide standard for publishing and querying trust information, this would be a prohibitively complex leading to verifiers having to deal with a high number of formats and protocols.

The EU-funded LIGHTest project attempts to solve this problem by building a global trust infrastructure where arbitrary authorities can publish their trust information. Setting up a global infrastructure is an ambitious objective; however, given the already existing infrastructure, organization, governance and security standards of the Internet Domain Name System, it is with confidence that this is possible. The EC and Member States can use this to publish lists of qualified trust services, as business registrars and authorities can in health, law enforcement and justice. In the private sector, this can be used to establish trust in inter-banking, international trade, shipping, business reputation and credit rating. Companies, administrations, and citizens can then use LIGHTest open source software to easily query this trust information to verify trust in simple signed documents or multi-faceted complex transactions.

The three-year LIGHTest project starts on September 1st and has an estimated cost of almost 9 Million Euros. It is partially funded by the European Union's Horizon 2020 research and innovation programme under G.A. No. 700321. The LIGHTest consortium consists of 14 partners from 9 European countries and is coordinated by Fraunhofer-Gesellschaft. To reach out beyond Europe, LIGHTest attempts to build up a global community based on international standards and open source software.

The partners are ATOS (ES), Time Lex (BE), Technische Universität Graz (AT), EEMA (BE), G+D (DE), Danmarks tekniske Universitet (DK), TUBITAK (TR), Universität Stuttgart (DE), Open Identity Exchange (GB), NLNet Labs (NL), CORREOS (ES), IBM Danmark (DK) and Globalsign (FI). The Fraunhofer IAO provides the vision and architecture for the project and is responsible for both, its management and the technical coordination.
The Fraunhofer IAO provides the vision and architecture for the project and is responsible for both, its management and the technical coordination.