Digital TRUST Infrastructure for Discovery and Validation
Technical and Functional Specification v0.5

Status: Public Review Draft
Authors: John Walker (john.walker1@undp.org), Lucy Yang (yang.qixue@undp.org)
Contributors: Savita Farooqui, Kendall Helms, Isaac Henderson, Juan Vargas

Table of Content

Project Introduction

The Digital TRUST Infrastructure for Discovery and Validation (DigiTRUST) project, sponsored by and hosted at the United Nation Development Programme (UNDP), is intended to develop and provide a suite of tools to enable the discovery and validation of trusted services by leveraging existing Internet infrastructures of the Domain Name System (DNS) and its security extensions.

Implementers of DigiTRUST can leverage the tools to create purpose-sized networks or ecosystems of trusted digital services operated by all types of entities. As a result, users will be able to easily discover services in need, access relevant service information through trusted endpoints and make informed decisions about whether to trust and use a service. Most importantly, DigiTRUST can enable all of this at a massive scale that today's fully centralized model is not able to. Thanks to the decentralized and cloud-agnostic architecture DigiTRUST adopts, any participating service of an implemented network or ecosystem on DigiTRUST will be able to maintain the sovereignty and control of their own systems and data. Such an approach provides the necessary trust infrastructure that can help thwart the ubiquitous phishing attempts mimicking online service organizations, such as government institutions, health providers and banks.

This document represents the specification of the technical and functional capabilities of DigiTRUST. It is the main specification document for DigiTRUST with additional supporting documents linked in appropriate sections.

Project Background

DigiTRUST was initially started at the Linux Foundation in 2021 as the Global COVID-19 Certificate Network (GCCN) to address the trust interoperability among COVID certificate networks or ecosystems. The project transitioned to UNDP in Summer 2022 with an added focus on developing governance and policy guidelines for the technical infrastructure. Since the transition, the UNDP team has been focusing on supporting the World Health Organization (WHO) to implement the G20 pilot for COVID Certificates while working to broaden the scope of the project to support digital health services and beyond. As we release the draft technical specification for public review, our team is working on an open source reference implementation for the specification, a governance and policy guideline for implementing DigiTRUST, and delivering GCCN as the first pilot and real-world use case for DigiTRUST.

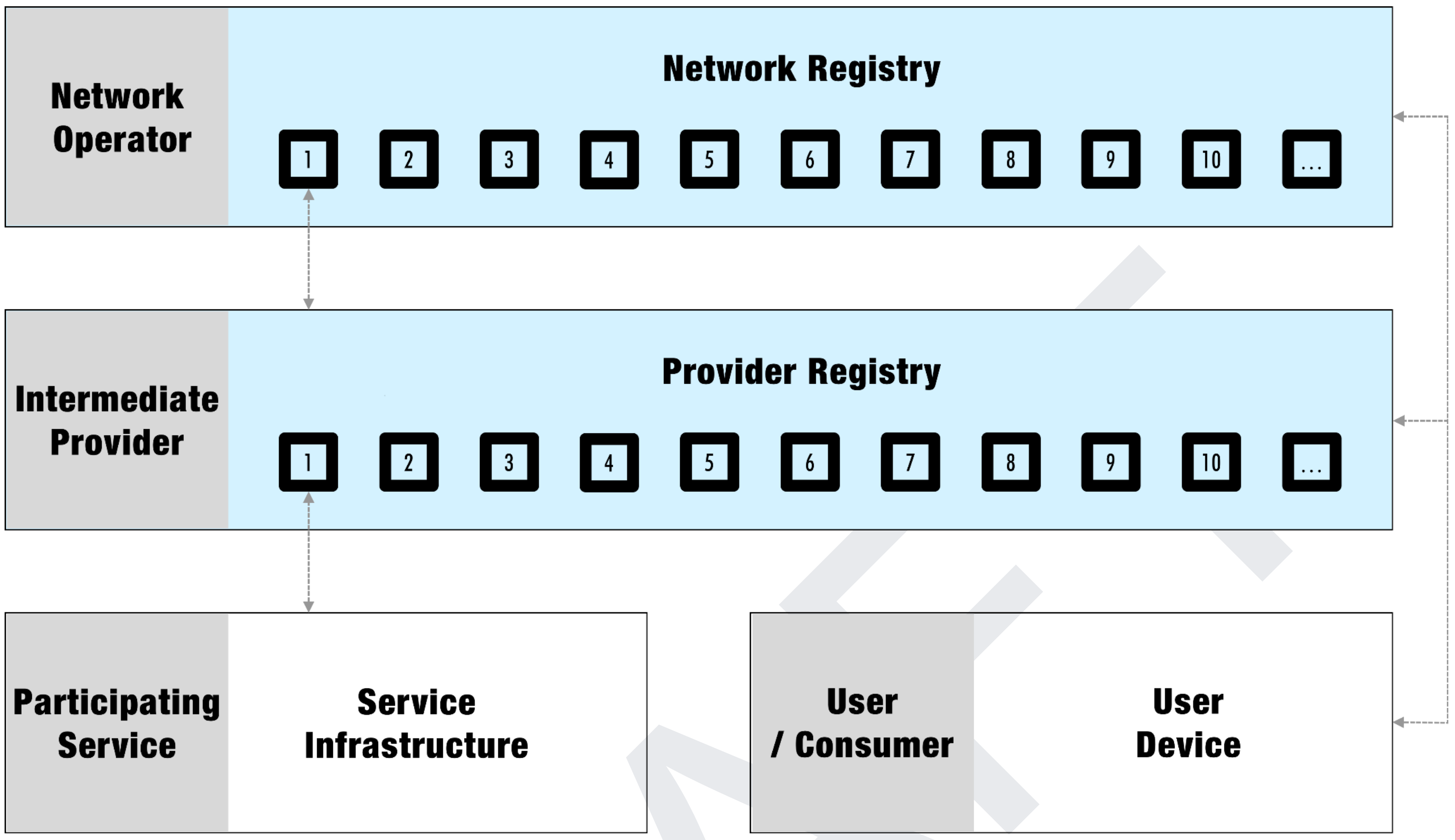
DigiTRUST is built on the open-source TRust mAnagement INfrastructure (TRAIN), developed by the Fraunhofer IAO, a world-leading applied research institute for industry engineering, as a part of the European Self-Sovereign Identity Framework (ESSIF) Lab. We chose TRAIN as the underlying technology for DigiTRUST due to its use/support of established standards and flexibility to work with existing architectures, which provides great technical accessibility and inclusivity for governments, organizations and companies across the world, regardless of their existing financial resources and technical capabilities. Some of its prominent features include:

- Provides agnostic support for all types of 'Root of Trust' architecture, e.g. X.509 certificate, Verifiable Credentials, Decentralized Identifier (DID)
- Supporting self-defined "Trust Schemes" and trust policies
- Supporting Trust Standards and Trust Schemes such as eIDAS, Pan Canadian Trust Framework
- Adopting the Trust Schema pattern from the ETSI 119 612 TS scheme for trust lists
- Providing the option to use Domain Name System Security Extensions (DNSSEC) to secure the chain of authenticity for service provider at different level of the implemented network/ecosystem/platform
- Providing an API for discovery and verification of trusted endpoints

Functional and Technical Specification

Operational and Technical Component Architecture

Operational Component Architecture



1. **Network Operator and Network Registry**

   Network Registry is the highest level (referred to as "Network Level") network or ecosystem of trusted digital service providers who participate in the discovery and validation mechanism run by a **Network Operator** implementing DigiTRUST.

2. **Intermediate Provider and Provider Registry**

   Intermediate Providers are the trusted digital service providers directly participating in the Network Registry. They may have their own network or ecosystem of trusted digital services, and as such can create their own **Provider Registry** by implementing DigiTRUST as the Intermediate Level.

   Note: For the purpose of developing this specification, we simplify the real-world implementations of DigiTRUST to two tiers of registry as indicated by the above operational architecture. In reality, there could be multiple layers of providers between the Network Registry and Participating Service that leverage the DigiTRUST technology. We also didn't project the scenario where an Intermediate Provider participates in the Network Registry without implementing DigiTRUST. We have taken considerations of these complicated operational scenarios into the development of the DigiTRUST technology and designed it to support such complications.

3. **Participating Service and Service Infrastructure**

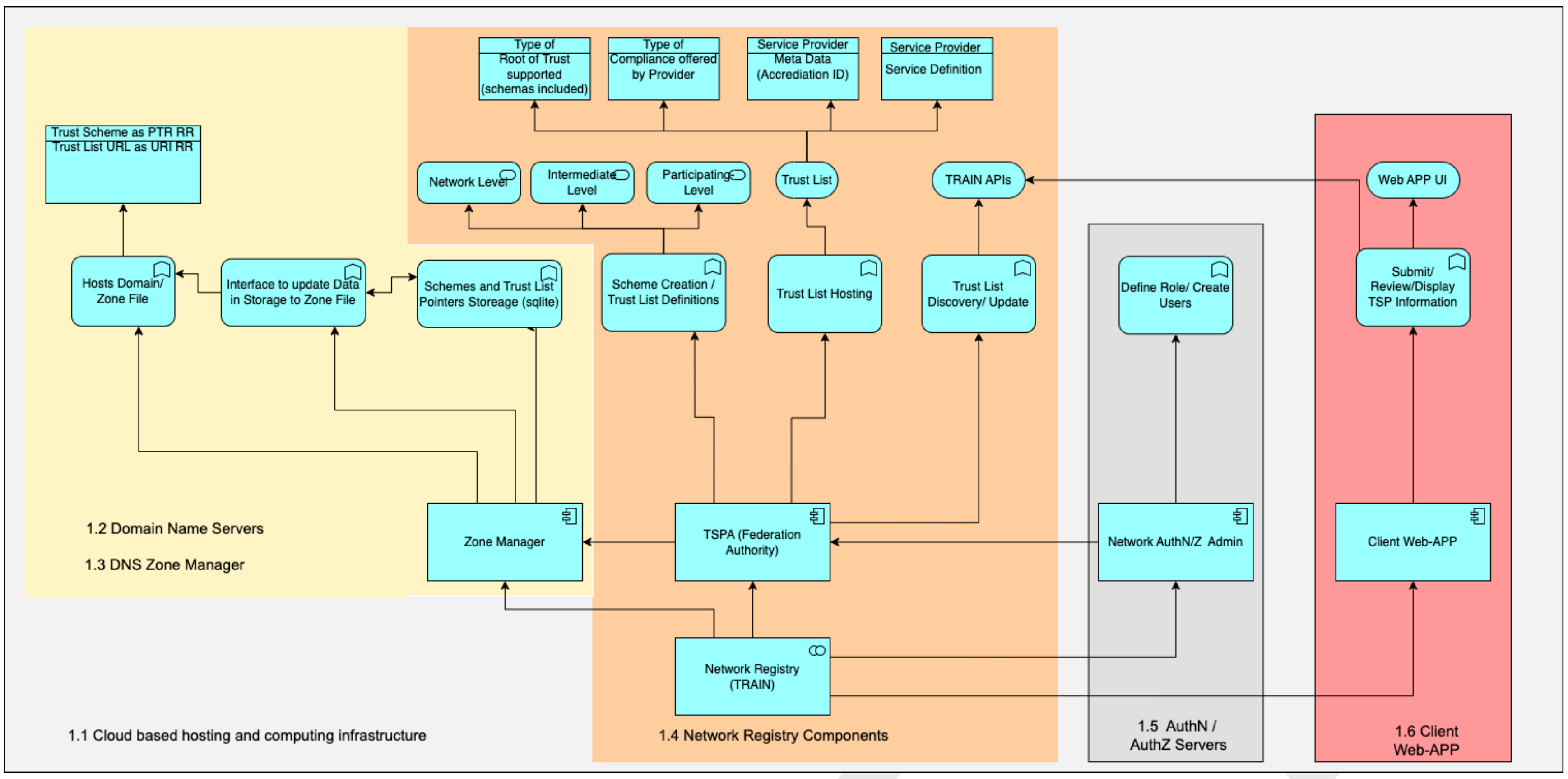
   Participating Service is the trusted digital service provider on the Provider Registry that leverages its own **Service Infrastructure**. Settings and activities related to the Participating Service or its Service Infrastructure are referred to as "Participating Level".

4. **User / Consumer and User Device**

   User / Consumer refers to any person or organization using the Network Registry for discovery and validation of trusted digital services. They would need to have some kind of **User Device** (e.g. mobile phone, laptop) to interact with the Network Registry.

Technical Component Architecture

1. Architectural Diagram and Technical Components


Type of Root of Trust supported (schemas included)
Type of Compliance offered by Provider
Service Provider Meta Data (Accreditation ID)
Service Provider Service Definition
Trust Scheme as PTR RR Trust List URL as URI RR
Network Level
Intermediate Level
Participating Level
Trust List
TRAIN APIs
Web APP UI
Hosts Domain/ Zone File
Interface to update Data in Storage to Zone File
Schemes and Trust List Pointers Storeage (sqlite)
Scheme Creation / Trust List Definitions
Trust List Hosting
Trust List Discovery/ Update
Define Role/ Create Users
Submit/ Review/Display TSP Information
1.2 Domain Name Servers
1.3 DNS Zone Manager
Zone Manager
TSPA (Federation Authority)
Network AuthN/Z Admin
Client Web-APP
Network Registry (TRAIN)
1.1 Cloud based hosting and computing infrastructure
1.4 Network Registry Components
1.5 AuthN / AuthZ Servers
1.6 Client Web-APP

Technical Component Architecture for the Network Registry depicted in Archimate 3.0 / Application

1.1. Cloud-based hosting and computing infrastructure

1.2. DNS name servers 🟨

1.3. DNS Zone Manager 🟨

1.4. Network Registry components 🟧

1.5. AuthN / AuthZ servers ⬜

1.6. Client Web App 🟥

We intend to add the technical component architecture for the Provider Registry in a later version of the specification - it will follow a similar pattern to the Network Registry architecture.

2. Key Technical Definitions and Roles

1.1. **Trust Scheme**: The structured definition of the entities and services represented in a data schema.

1.2. **Trust List:** A list of all the service providers and their respective service definitions that are defined in a specific Trust Scheme.

1.3. **Trust Service Publication Authority (TSPA):** An entity authorized to define the schema of a Trust Scheme and publish Trust Schemes including the terminology and policies available for a network or ecosystem implementing DigiTRUST

1.4. **Trust Publication Authority (TPA)**: An entity appointed by the TSPA to enroll services adhering to the respective Trust Scheme into a Trust List and publish and maintain the Trust List. Sometimes it is the same entity as the TSPA.

1.5. **Trust Service Provider (TSP):** A service provider enrolled in a Trust List by a TPA

Operational Actor Roles

Operational Actor Roles are the distinct roles needed at the operational level. Some of these operational roles either individually or collectively fulfill the technical roles defined in the previous section. At each operational level, the same entity/person can play multiple roles at the same time.

1. **Network Operator (Performing the technical roles of TSPA and TPA at the Network Level)**

    1.1. **Network Super Administrator:** This role instantiates the Network Registry cloud Infrastructure components, the authentication (AuthN) and authorization (AuthZ) infrastructure and the functional components of the Network Registry.

    1.2. **Network Realm Administrator:** This role is responsible for creating Network Registry user accounts, and assigning them the various roles within the application.

    1.3. **Network Entry Approver:** This role is responsible for reviewing the submitted Intermediate Providers' TSPA profiles and approving them to become entries on the Network Registry (or rejecting them).

2. **Intermediate Provider (Performing the technical roles of TSPA and TPA at the Intermediate Level)**

    2.1. **Provider Registry Administrator:** This role instantiates the Provider Registry components, the authentication (AuthN) and authorization (AuthZ) infrastructure, and the functional components of the Provider Registry.

    2.2. **Provider Realm Administrator:** This role is responsible for creating Provider Registry user accounts, and assigning them the various roles within the application.

    2.3. **Network Entry Submitter:** This role is responsible for submitting the Intermediate Provider's TSPA profile to participate in the Network Registry.

    2.4. **Provider Entry Approver:** This role is responsible for reviewing the submitted Participating Services' TSP profiles and approving them to become entries on the Provider Registry (or rejecting them), thus making the Participating Services discoverable through the Network Registry.

3. **Participating Service (Performing the technical role of TSP)**

    3.1. **Provider Entry Submitter:** This role is responsible for submitting the Participating Service's TSP profile to Provider Registries it wants to participate in.

4. **User / Consumer**

    4.1. **Network Registry Consumer:** The role is applicable to any person or organization who wishes to discover and validate trusted Intermediate Providers and their Participating Services on a Network Registry.

Functional Scope: Functional Description of Actors and System Capabilities

We provide the use cases for the two tiers of implementations of DigiTRUST to provide a high level overview of the key capabilities of the technology. The actual implementation and business processes need to follow the operational and governance guidance defined by a Network Operator, likely in collaboration with the Intermediate Providers.

Network Registry

1. **Use Case: Instantiate the Network Level infrastructure**

   The Network Registry is a cloud-based infrastructure that requires establishing a <network name>.<network_operator> DNS entry and the instantiation of the supporting 'Technical Components'.

   1.1. **Actor:** Network Super Administrator

   1.2. **Pre-condition:** A cloud network environment has been identified and access for a system administrator to that cloud project space with the necessary admin rights has been provided.

   1.3. **Basic Flow:** The Network Super Administrator **SHALL** instantiate and provide administrator user access to

   1.3.1. Cloud environment: Network Level project space, IP network, host compute resources, specified operation system and environment software utility installs.
   - [Referred to as: Network Level Base Cloud Environment]

   1.3.2. Instantiate DNS domain "<network name>.<network_operator>.<TLD extension>"

   1.3.3. Primary and secondary name servers -
   - Base cloud environment
   - Domain server installation per 'DNS Configuration' of this document
   - Bind Network Level name servers to the IP Host addresses

   1.3.4. Zone Manager
   - Base cloud and DNS environment
   - Zone Manager installation per 'DNS Configuration' of this document
     - [Together with Network Registry 1.3.3, referred to as: Network Level Base DNS Environment]

   1.3.5. AuthN and AuthZ Servers
   - Base cloud and DNS environment
   - KeyCloak installation per 'AuthN / AuthZ Servers Installation and Configuration' of this document
     - [Referred to as: Network Level base Auth (N and Z) environment]

   1.3.6. TRAIN TSPA Host
   - Base cloud, DNS and Auth environment
   - TRAIN TSPA Infrastructure installation per 'TRAIN API and Host Requirements' of this document

   1.3.7. TRAIN API Host
   - Base cloud, DNS and Auth environment
   - TRAIN API Host Infrastructure installation per 'TRAIN API and Host Requirements' of this document
     - [Together with Network Registry 1.3.6, referred to as: Network Level Base TRAIN Environment]

   1.3.8. Client Web App build and deployment
   - Base cloud, DNS, Auth and TRAIN environment
   - Network Level client infrastructure installation per 'Client Web App Configuration' of this document
     - [Referred to as: Network Level Base Client Environment]

   1.4. **Post Condition:** Network access to all hosts and installed infrastructure and applications listed in 'Technical Components' is available for login by the Network Super Administrator

   1.4.1. The successful installation of all the 'Technical Components' **SHALL** be referred to as a 'Network Level Base Operating Environment'

2. **Use Case: Configure the Network Level infrastructure instance**

   The Network Super Administrator **SHALL** configure the Network Level Base Operating Environment per the specific cloud hosting environment selected and the configuration requirements of the Network Operator.

2.1. **Actor:** Network Super Administrator

2.2. **Pre-condition:** A 'Network Level Base Operating Environment' - installation and network availability for a Network Super Administrator, has been successfully completed.

2.3. **Basic flow:** The Network Super Administrator **SHALL** configure the Network Level resources -

2.3.1. Add the Network Realm Administrator, Network Entry Approver and Network Entry Submitter roles
- Import Network Level realm file
- Add / Edit role definitions per the Network Operator requirements
- Configure the Network Level KeyCloak client
- Add user(s) and associate with the Network Realm Administrator role

2.3.2. Configure the DNS name servers per the Network Operator requirements and 'DNS Configuration' of this document

2.3.3. Configure the DNS Zone Manager per the Network Operator requirements and 'DNS Configuration' of this document

2.3.4. Instantiate the Network Operator Trust List
- Update the <SchemeInformation> properties section in the network-registry.<network-operator>.trust-scheme.en XML per the requirements of the Network Operator
- Publish the network-registry.<network-operator>.trust-scheme.en XML via TRAIN
- Confirm (via a web browser) that the network-registry.<network-operator>.trust-scheme.en XML resolves to its specified URL.

2.4. **Post Condition:** The Network Level (KeyCloak) realm, client settings, and authorization roles have been configured. The Network Level DNS name servers have been configured and the Network Operator domain is established. The Network Level Trust Scheme has been updated and instantiated.

3. Use Case: Add a Network Entry Approver to the Network Level realm

The Network Realm Administrator(s) **SHALL** add the Network Entry Approver(s) to as Network Level users by assigning the designated users to a Network Entry Approver role in the Network Level realm.

3.1. **Actor:** Network Realm Administrator

3.2. **Pre-condition:** The availability of the Network Level AuthN / AuthZ server, and an instantiated network-registry.<network-operator>.trust-scheme.en.

3.3. **Basic Flow:** The Network Realm Administrator add a Network Entry Approver to the Network Level AuthN / AuthZ server

3.4. **Post Condition:** A Network Entry Approver has been added to the Network Level AuthN / AuthZ server and can be viewed in the KeyCloak console. The Network Entry Approver can log in to the Network Level client software.

4. Use Case: Add a Network Entry Submitter to the Network Level realm

The Network Realm Administrator(s) **SHALL** add Intermediate Providers' Network Entry Submitters as Network Level users by assigning the designated users to a Network Entry Submitter role in the Network Level realm.

4.1. **Actor:** Network Realm Administrator

4.2. **Pre-condition:** The availability of the Network Level AuthN / AuthZ server, and an instantiated network-registry.<network-operator>.trust-scheme.en.

4.3. **Basic Flow:** The Network Realm Administrator add a Network Entry Submitter to the Network Level AuthN / AuthZ server

4.4. **Post Condition:** A Network Entry Submitter role to the Network Level AuthN / AuthZ server has been added and can be viewed in the KeyCloak console. The Network Entry Submitter(s) can log in to the Network Level client software.

5. Use Case: Submit an Intermediate Provider's TSPA profile to the Network Registry

The Network Entry Submitter(s) **SHALL** submit their Intermediate Providers' TSPA profiles to the Network Registry by completing required client screen forms and submitting the completed forms via the client interface to the Network Registry for review.

5.1. **Actor:** Network Entry Submitter

5.2. **Pre-condition:** Network Registry availability, Network Level client software access, configured AuthN / AuthZ server, and an instantiated network-registry.<network-operator>.trust-scheme.en.

5.3. **Basic Flow:** The Network Entry Submitter **SHALL** submit its Intermediate Provider's TSPA complete profile to the Network Registry for approval via the client software, by completing the multi-page form and using the "Submit" button at the bottom of the "Service Operational Contact Information" page.

5.4. **Post Condition:** Successful submission of a Intermediate Provider's TSPA profile

4.4.1. Email notification of the submission will be sent to the Network Registry Approver for review
4.4.2. The submitted TSPA profile is marked as "Pending"

6. Use Case: Edit an Intermediate Provider's TSPA profile prior to approval

The Network Entry Submitter(s) **MAY** edit their submitted TSPA profiles in a "Pending" state listed on the "My Submissions" page via the Network Level client software. Edited profiles are re-submitted by using the "Save and Exit" button at the bottom of each form page.

6.1. **Actor:** Network Entry Submitter

6.2. **Pre-condition:** Network Registry availability, Network Level client software access, and configured AuthN / AuthZ server.

6.3. **Basic Flow:** The Network Entry Submitter updates and submits its Intermediate Provider's TSPA profile to the Network Registry for approval via the client software, using the 'Save and Exit' button at the bottom of any page of the form to complete the process.

6.4. **Post Condition:** Upon successful update and submission of the Intermediate Provider's TSPA profile, email notification of the change and submission will be sent to a Network Registry Approver for review.

7. Use Case: Review and Reject an Intermediate Provider's TSPA profile submission

The Network Registry Approver **SHALL** review the submitted Intermediate Providers' TSPA profiles via the Network Level client's Network Entry Submissions List page with an entry Status= 'PENDING'; completion of the review ends with submitted completed forms having a 'Rejected' status applied to the submission by the Network Registry Approver using the Network Entry Submission page "Reject Submission" button.

7.1. **Actors:** Network Entry Approver

7.2. **Pre-condition:** Network Registry availability, Network Level client software access, configured AuthN / AuthZ server.

7.3. **Basic Flow:** The Network Entry Approver selects a submitted Intermediate Provider's TSPA profile, reviews it against business rules established by the Network Operator and marks it as 'Rejected' via the client software.

7.3.1. TSPA submission form review: Properties are reviewed in accordance to business rules determined by the Network Operator for TSPA submissions. Individual properties on the forms will be reviewed for acceptance as an offline process.
7.3.2. The Network Registry Approver verified entry, with a "Rejected" status, **SHALL** have:

- Email notification to the Intermediate Provider's Network Entry Submitter of the rejection and request for edits or clarification.
- The Network Registry Approver MAY establish an offline review of the submitted profile with the Network Entry Submitter to resolve the issues with the submitted profile.
- The Network Registry Approver MAY apply edits as a result of an offline review process; the edited profile is re-submitted with a 'Pending' state.

7.4. **Post Condition:** A "Reject Submission" action generates an email notification of the rejection and MAY initiate offline mitigation to correct any errors in the submission.

8. Use Case: Review and Accept an Intermediate Provider's TSPA profile submission

The Network Registry Approver(s) **SHALL** review the submitted Intermediate Providers' TSPA profiles via the Network Level client's Network Entry Submissions List page with an entry Status= 'PENDING'; completion of the review ends with submitted completed forms having an 'Approved' status applied to the submission by the Network

Registry Approver using the Network Entry Submission page “Accept Submission” button.

8.1. **Actors:** Network Entry Approver

8.2. **Pre-condition:** Network Registry availability, Network Level client software access, configured AuthN / AuthZ server.

8.3. **Basic Flow:** The Network Entry Approver selects a submitted Intermediate Provider’s TSPA profile, reviews it against business rules established by the Network Operator and marks it as ‘Accepted’ based on a successful review via the client software

8.3.1. TSPA submission form review: Properties **SHALL** be reviewed in accordance to business rules determined by the Network Operator for TSPA submissions. Individual properties on the forms will be reviewed for acceptance as an offline process.

8.3.2. The Network Registry Approver verified entry, with an “Accepted” status **SHALL** be automatically submitted from the Network Level client software to the TRAIN API for insertion into the Network Registry.

8.4. **Post Condition:** Submission of the Intermediate Provider’s TSPA profile resulting in an “Accept Submission” will have an “Accepted” status and be inserted into the Network Registry.

9. **Use Case: Insert an accepted Intermediate Provider’s TSPA profile into the Network Registry (back-end)**

9.1. **Actors:** Network Entry Approver, Network Super Administrator

9.2. **Pre-condition:** Network Registry availability, Network Level client software access, configured AuthN / AuthZ server, and an approved Intermediate Provider’s TSPA profile.

9.3. **Basic Flow:** The Network Entry Approver selects ‘Accept Submission’ button.

9.3.1. Network Level client software executes a “PUT” method against the TRAIN API and successful submission returns a “Successfully Submitted” message via the Network Level client software

9.4. **Alternate Flow:** Network Level client software executes a “PUT” method against the TRAIN API and submission failure returns a “Submission Error” message via the Network Level client software.

9.4.1. Network Super Administrator is emailed with the “Submission Error” message and investigates TRAIN error logging to triage and re-submit the profile.

9.5. **Post Condition:** The Intermediate Provider’s TSPA profile is inserted into the network-registry.<network-operator>.trust-scheme.en XML file and is a resolvable URL at “https://<network-domain>/tspa/api/v1/network-name/networkregistry/{TrustSchemeName}”

Provider Registry

The implementations of Provider Registries follow the pattern established for the ‘Network Registry’. Accordingly, use cases start with the infrastructure instantiation, executed by the Intermediate Provider’s administrators, and completes with the insertion of Participating Services’s TSP profiles into their respective Intermediate Providers’ Trust Schemes.

1. **Use Case: Instantiate the Provider Registry infrastructure**

The Provider Registry is a cloud-based infrastructure that requires establishing a <provider name>.<intermediate_provider> DNS entry and the instantiation of supporting the ‘Technical Components’.

1.1. **Actor:** Provider Registry Administrator

1.2. **Pre-condition:** A cloud network environment has been identified and access for a system administrator to that cloud project space with the necessary admin rights has been provided.

1.3. **Basic Flow:** The Provider Registry Administrator **SHALL** instantiate and provide administrator user access to -

1.3.1. Cloud environment: Intermediate Level project space, IP network, host compute resources, specified operating system and environment software utility installs.

- [Referred to as: Intermediate Level Base Cloud Environment]

1.3.2. Instantiate Intermediate Level DNS domain “<provider name>.<intermediate_provider>.<TLD

extension>"

1.3.3. Primary and Secondary name servers -
- Base cloud environment
- Domain server installation per 'DNS Configuration' of this document
- Bind Intermediate Level name servers to the IP Host addresses

1.3.4. Zone Manager
- Base cloud and DNS environment
- Zone Manager installation per 'DNS Configuration' of this document
  - [Together with Provider Registry 1.3.3., referred to as: Intermediate Level Base DNS environment]

1.3.5. KeyCloak Auth N and Z server
- Base cloud and DNS environment
- KeyCloak installation per 'AuthN / AuthZ Servers Installation and Configuration' of this document
  - [Referred to as: Intermediate Level Base Auth (N and Z) Environment]

1.3.6. TRAIN TSPA Host
- Base cloud, DNS and Auth environment
- TRAIN TSPA Infrastructure installation per 'TRAIN API and Host Requirements' of this document

1.3.7. TRAIN API Host
- Base cloud, DNS and Auth environment
- TRAIN API Host Infrastructure installation per 'TRAIN API and Host Requirements' of this document
  - [Together with Provider Registry 1.3.6., referred to as: Intermediate Level Base TRAIN Environment]

1.3.8. Client Web App build and deployment
- Base cloud, DNS, Auth and TRAIN environment
- Client infrastructure installation per 'Client Web App Configuration' of this document
  - [Referred to as: Intermediate Level Base Client Environment]

1.4. **Post Condition:** Network access to all hosts and installed infrastructure and applications listed in 'Technical Components' is available for login by the Provider Registry Administrator.

1.4.1. The successful installation of all the 'Technical Components' **SHALL** be referred to as a 'Intermediate Level Base Operating Environment'

2. Use Case: Configure the Intermediate Level infrastructure instance

The Provider Registry Administrator **SHALL** configure the Intermediate Level Base Operating Environment per the specific cloud hosting environment chosen and the configuration requirements of the Intermediate Provider.

2.1. **Actor:** Provider Registry Administrator

2.2. **Pre-condition:** A 'Intermediate Level Base Operating Environment' - installation and network availability for a Provider Registry Administrator, has been successfully completed.

2.3. **Basic flow:** The Provider Registry Administrator **SHALL** configure the Intermediate Level resources -

2.3.1. Add the Provider Realm Administrator, Provider Entry Approver, and Provider Entry Submitter roles
- Import Intermediate Level realm file
- Add / Edit role definitions per the Intermediate Provider requirements
- Configure the KeyCloak client
- Add users and associate with the Provider Realm Administrator role

2.3.2. Configure the DNS name servers per the Intermediate Provider requirements and 'DNS Configuration' of this document

2.3.3. Configure the DNS Zone Manager per the Intermediate Provider requirements and 'DNS Configuration' of this document

2.3.4. Instantiate the Intermediate Provider Trust List
- Update the <SchemeInformation> properties section in the provider-registry.<intermediate-provider>.trust-scheme.en XML per the requirements of the Intermediate Provider
- Publish the provider-registry.<intermediate-provider>.trust-scheme.en XML via TRAIN
- Confirm (via a web browser) that the provider-registry.<intermediate-provider>.trust-scheme.en XML resolves to its specified URL.

2.4. **Post Condition:** The Intermediate Level (KeyCloak) realm, client settings, and authorization roles are configured. The Intermediate Level DNS name servers have been configured and the Intermediate Provider domain is established. The Intermediate LevelTrust Scheme has been updated and instantiated.

3. Use Case: Add a Provider Entry Approver to the Intermediate Level realm

The Provider Realm Administrator(s) **SHALL** add the Provider Entry Approver(s) as Intermediate Level users by assigning the designated user(s) as a Provider Entry Approver role in the Intermediate Level realm.

3.1. **Actor:** Provider Realm Administrator

3.2. **Pre-condition:** The availability of the Intermediate Level AuthN / AuthZ server, and an instantiated provider-registry.<intermediate-provider>.trust-scheme.en.

3.3. **Basic Flow:** The Provider Realm Administrator add a Provider Entry Approver to the the Intermediate Level AuthN / AuthZ server

3.4. **Post Condition:** A Provider Entry Approver has been added to the Intermediate Level AuthN / AuthZ server and can be viewed in the KeyCloak console. The Provider Entry Approver can log in to the Intermediate Level client software.

4. Use Case: Add a Provider Entry Submitter to the Intermediate Level realm

The Provider Realm Administrator **SHALL** add Participating Services' Provider Entry Submitters as Intermediate Level users by assigning the designated user as a Provider Entry Submitter role in the Intermediate Level realm.

4.1. **Actor:** Provider Realm Administrator

4.2. **Pre-condition:** The availability of the Intermediate Level AuthN / AuthZ server, and an instantiated provider-registry.<intermediate-provider>.trust-scheme.en.

4.3. **Basic Flow:** The Provider Realm Administrator add a Provider Entry Submitter to the Intermediate Level AuthN / AuthZ server

4.4. **Post Condition:** A Provider Entry Submitter role to the Intermediate Level AuthN / AuthZ server has been added and can be viewed in the KeyCloak console. The Provider Entry Submitter can log in to the Intermediate Level client software.

5. Use Case: Submit a Participating Service's TSP profile to the Provider Registry

The Provider Entry Submitters **SHALL** add their Participating Services' TSP profiles to the Provider Registry by completing required multi-page forms and submitting the completed forms via the client interface to Provider Entry Approver for review.

5.1. **Actor:** Provider Entry Submitter

5.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, configured AuthN / AuthZ server, and an instantiated provider-registry.<intermediate-provider>.trust-scheme.en.

5.3. **Basic Flow:** The Provider Entry Submitter **SHALL** submit its Participating Service's TSP complete profile to the Provider Registry via the client software by completing the multi-page form and selecting the "Submit" button at the bottom of the "Service Operational Contact Information" page.

5.4. **Post Condition:** Successful submission of a Participating Service's TSP profile

4.4.3. Email notification of the submission will be sent to the Provider Entry Approver for review
4.4.4. The submitted TSP profile is marked as "Pending"

6. Use Case: Edit a Participating Service's TSP profile prior to approval

The Provider Entry Submitter(s) **MAY** edit their submitted TSP profiles in a "Pending" state listed on the "My Submissions" page via the Intermediate Level client software. Edited profiles are re-submitted by using the "Save and Exit" button at the bottom of each form page.

6.1. **Actor:** Provider Entry Submitter

6.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, and configured AuthN / AuthZ server.

6.3. **Basic Flow:** The Provider Entry Submitter

7.3.1. Review submissions on the Submissions List page
7.3.2. Select a submission to Edit from the list presented
7.3.3. Edit the profile via the Submission multi-page form
7.3.4. Complete and submit using the 'Save and Exit' button

6.4. **Post Condition:** Upon successful update and submission of the Participating Service's TSP profile, email notification of the change and submission will be sent to a Provider Registry Approver for review.

7. Use Case: Review a Participating Service's TSP profile submission

The Provider Entry Approver **MAY** review the submitted Participating Service's TSP profile via the Intermediate Level client software. The Provider Registry Approver will be redirected to the home page, which will now include the "Registry Entries" and "Review Submissions" links in the navigation bar.

7.1. **Actor:** Provider Entry Approver

7.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, and configured AuthN / AuthZ server.

7.3. **Basic Flow**: The Provider Entry Approver:

8.3.1. Views submissions in a "Pending" state on the Submissions List page
8.3.2. Selects a submission to Review from the list presented
8.3.3. Reviews the submission via that entries' multi-page form
8.3.4. Completes the review without a decision using the 'Save and Exit' button

7.4. **Post Condition:** Upon completion of a review of a Participating Service's TSP profile,

8.4.1. No change in the "Pending" status of the entry will be made
8.4.2. The Provider Entry Approver **MAY** return to their list "Pending" state on the Submissions List page to review any listed submission for review

8. Use Case: Review and Reject a Participating Service's TSP profile submission

The Provider Entry Approver(s) **MAY** review the submitted Participating Services' TSP profiles via the Intermediate Level client software. The Provider Entry Approver will be redirected to the home page, which will now include the "Registry Entries" and "Review Submissions" links in the navigation bar.

8.1. **Actor:** Provider Entry Approver

8.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, and configured AuthN / AuthZ server.

8.3. **Basic Flow:** The Provider Entry Approver:

8.3.1. Selects an entry in a "Pending" state on the Submissions List page
8.3.2. Review the entry by navigating the entries' multi-page form
8.3.3. Concludes the review by rejecting the submission by using the 'Reject and Exit' button

8.4. **Post Condition:** Upon rejection of a Participating Service's TSP profile submission:

8.4.1. The submission's status is changed to "Rejected" and this is reflected in its listed status
8.4.2. The Provider Entry Submitter is notified via email of the rejection.
8.4.3. The Provider Entry Approver **MAY** return to their list "Pending" state on the Submissions List page to review any listed submissions for review.
8.4.4. The Provider Entry Approver **MAY** engage the Service Entry Submitter to address and correct issues in the TSP profile that caused the rejection.

9. Use Case: Review and Accept a Participating Service's TSP profile submission

The Provider Entry Approver(s) **SHALL** approve the submitted Participating Services' TSP profiles after successful review of the profiles via the Intermediate Level client software. The Provider Entry Approvers will be redirected to the home page, which will now include the "Registry Entries" and "Review Submissions" links in the navigation bar.

9.1. **Actor:** Provider Entry Approver

9.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, and configured AuthN / AuthZ server.

9.3. **Basic Flow:** The Provider Entry Approver:

9.3.1. Selects a submission in a "Pending" state on the Submissions List page
9.3.2. Reviews the submission by navigating its multi-page form
9.3.3. Concludes the review by accepting the submission using the 'Accept and Exit' button

9.4. **Post Condition:** Upon acceptance of a Participating Service's TSP profile:

9.4.1. The submission's status is changed to "Accepted" and this is reflected in its listed status
9.4.2. The Service Entry Submitter is notified via email of the acceptance.
9.4.3. The Provider Entry Approver **MAY** return to their list "Pending" state on the Submissions List page to review any listed submission for review.
9.4.4. The Provider Entry Approver **MAY** engage the Service Entry Submitter to instigate any follow on processes necessary to initiate its availability online.

10. Use Case: Insert an accepted Participating Service's TSP profile into the Provider Registry

10.1. **Actors:** Provider Entry Approver, Provider Registry Administrator

10.2. **Pre-condition:** Provider Registry availability, Intermediate Level client software access, configured AuthN / AuthZ server, and an approved Participating Servicer's TSP profile.

10.3. **Basic Flow:** The Provider Entry Approver selects 'Accept Submission' button.

10.3.1. Intermediate Level client software executes a "PUT" method against the TRAIN API and successful submission returns a "Successfully Submitted" message via the Intermediate Level client software

10.4. **Alternate Flow:** Intermediate Level client software executes a "PUT" method against the TRAIN API and submission failure returns a "Submission Error" message via the Intermediate Level client software.

10.4.1. Provider Registry Administrator is emailed with the "Submission Error" message and investigates TRAIN error logging to triage and re-submit the profile.

10.5. **Post Condition:** The Participating Servicer's TSP profile is inserted into the provider-registry.<intermediate-provider>.trust-scheme.en XML file and is a resolvable URL at "https://<network-domain>/tspa/api/v1/network-name/providerregistry/{TrustSchemeName}"

DNS Configuration

Basic DNS Configuration

1. **IETF compliant Domain Name Server implementation**

   The DNS configuration works with IETF compliant Domain Name Server implementation

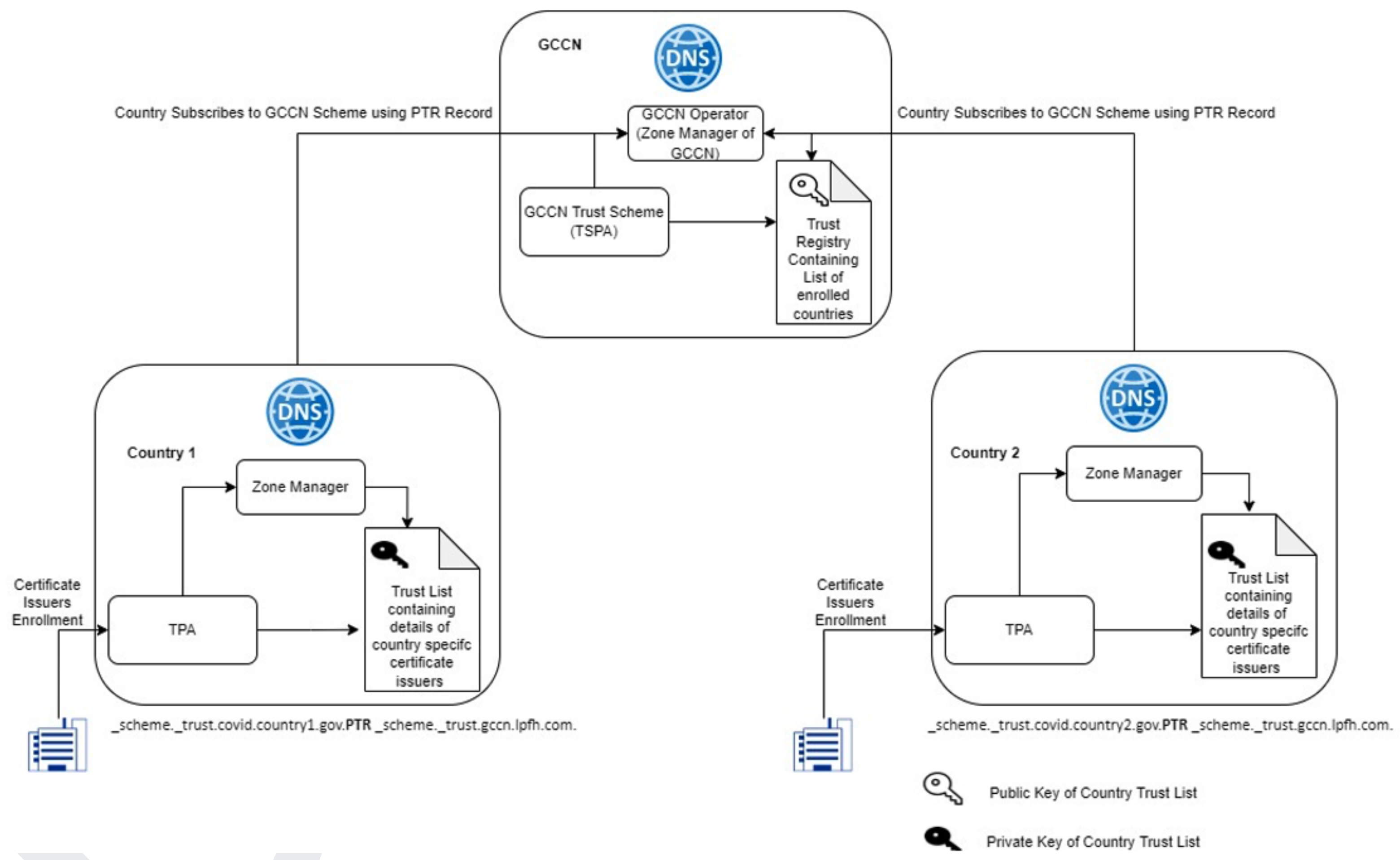
2. **Two-tier implementation of DNS zones**

   The DNS configuration is based on a two-tier implementation of DNS zones, the top tier (referred to as the Network Registry), is the root domain for the Network Level instance. The proposed domain naming pattern is, “<network name>.<network-operator>.<TLD>” .

   2.1. **Network Level Domain:** The Network Level domain definition consists of the qualified domain name which references the Network Level XML list of Intermediate Providers, AND IP Host addresses, ‘A’ records referencing each respective participating sub-domain.

   2.2. **Intermediate Level Domain:** The Intermediate Level sub-domains utilize a Zone Manager with PTR (reverse DNS) records to refer their IP host addresses back to the Network Level Zone Manager.

   2.2.1 The LIGHTest Zone Manager daemon software is used to instantiate the DNS domain and zone manager records.


GCCN
DNS
Country Subscribes to GCCN Scheme using PTR Record
GCCN Operator (Zone Manager of GCCN)
Country Subscribes to GCCN Scheme using PTR Record
GCCN Trust Scheme (TSPA)
Trust Registry Containing List of enrolled countries
DNS
Country 1
Zone Manager
Certificate Issuers Enrollment
TPA
Trust List containing details of country specifc certificate issuers
_scheme._trust.covid.country1.gov.PTR _scheme._trust.gccn.lpfh.com.
DNS
Country 2
Zone Manager
Certificate Issuers Enrollment
TPA
Trust List containing details of country specifc certificate issuers
_scheme._trust.covid.country2.gov.PTR _scheme._trust.gccn.lpfh.com.
Public Key of Country Trust List
Private Key of Country Trust List

Zone Manager

The configuration will follow the configuration pattern established for the LIGHTest Zone Manager demonstration environment. The TRAIN Trust Scheme management component leverages this DNS configuration for the instantiation and updates to the DNS-rooted Trust Scheme XML files.

1. **The LIGHTest configuration**

   https://github.com/H2020LIGHTest/ZoneManager/blob/master/doc/install-demo.md

2. **The DNS servers being utilized are NLNET Labs NSD servers version 4.6.0**

   2.1. NSD technical documentation https://nsd.docs.nlnetlabs.nl/en/latest/
   2.2. NSD Git Hub https://github.com/NLnetLabs/nsd

3. **NSD Implementation Manual**

   https://manpages.ubuntu.com/manpages/trusty/man8/nsd.8.html

AuthN / AuthZ Servers Installation and Configuration

KeyCloak will be instantiated on separate cloud hosts providing AuthN and AuthZ services for the implemented application and platform services. KeyCloak supports the OpenID Connect open protocol standard, OpenID Connect will be the authentication and authorization protocol utilized for the implemented solution platform access.

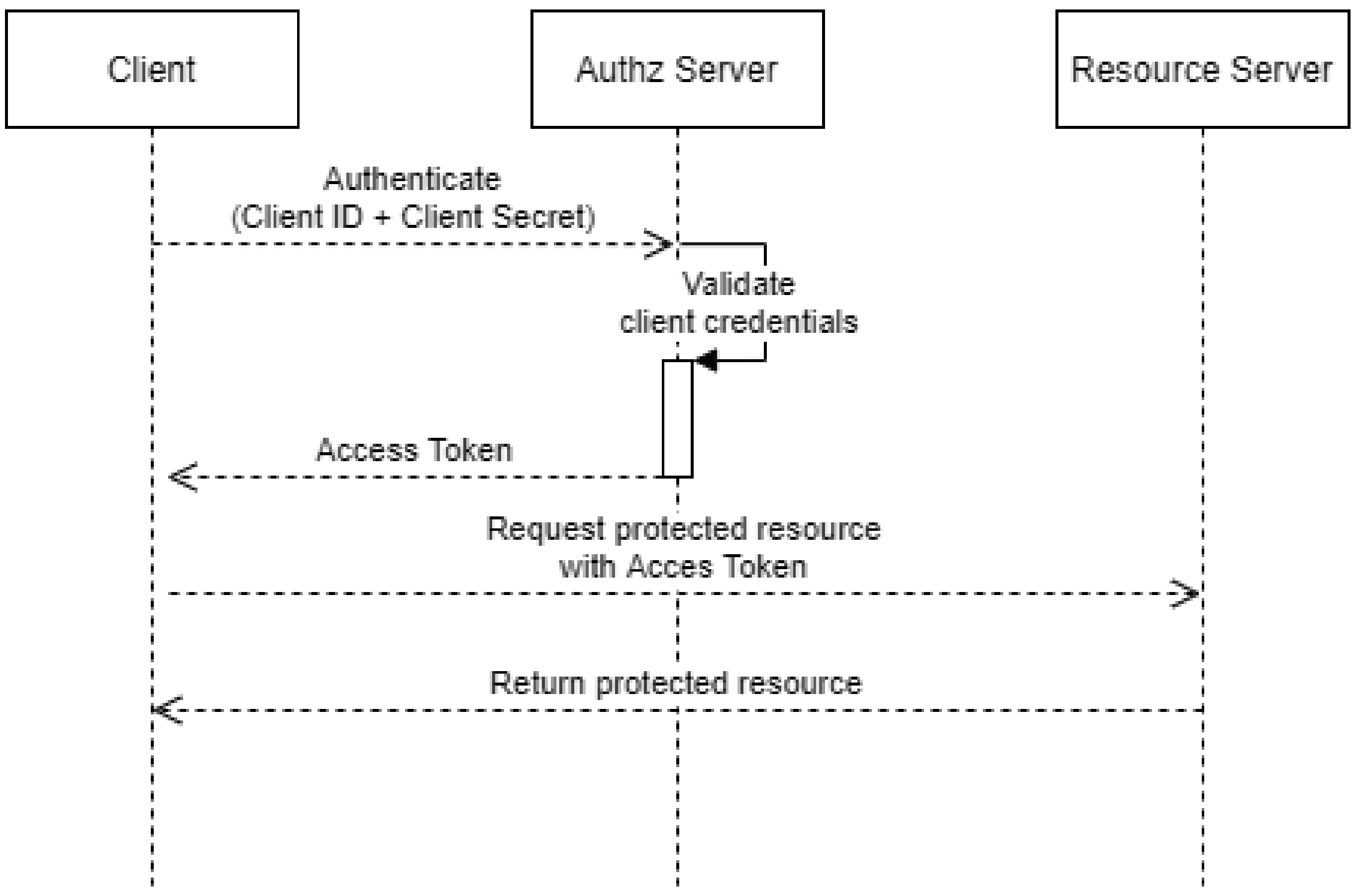
Install KeyCloak

Configure KeyCloak

1. **Base Development Configuration**

   1.1. Download the KeyCloak realm configuration 'Client Web App Configuration'.
   1.2. Through the command line, open the KeyCloak installation folder.
   1.3. Start KeyCloak using the command "bin\kc.bat start-dev".
   1.4. Open your web browser and enter the URL http://localhost:8080 – this will bring you to the KeyCloak Administration Console.
   1.5. Click the "Administration Console" link, and follow the prompts to create a master account for this KeyCloak instance.
   1.6. Once signed in to your master account, click the dropdown with the "Master" option selected in the top left corner of the screen, and click the "Create Realm" button.
   1.7. Now, from the "Create Realm" page, use the realm configuration file downloaded earlier as the resource file.
   1.8. Set the realm name to "xxx_Realm"
   1.9. Click the "Create" button. The KeyCloak realm is now configured.
   1.10. Users for each user role will need to be created within the new realm.
   1.11. Leave the command line open while running the application.

2. **Base KeyCloak AuthZ Token flow:**



Authorization Services Guide

https://www.KeyCloak.org/docs/latest/authorization_services/index.html

Base KeyCloak Realm File

https://github.com/undp/gccn/tree/main/auth/KeyCloak

TRAIN API and Host Requirements

TRAIN - **TR**ust m**A**nagement **IN**frastructure is the backend application component of the implemented solution responsible for submission of, and insertions and updates to the Network Registry and Provider Registry XML files. TRAIN exposes Public APIs and secured Bearer Token authorization.

API Methods

(Note: we will have new links to share in the official v1.0 of the specification and update terms of the APIs to reflect the new name of the project)

1. PUT - Publish Trust-List, publish the included TSP details

   Auth: Bearer Token
   Path: <Train_Base_URL>/tspa/api/v1/network-name/trustlist/publish/{TrustSchemeName}
   Payload: TSP Detail [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]
   Return Codes: [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]

2. GET - Fetch Trust-List , fetch the list of TSPA entries on the Network Registry

   Auth: Public
   Path: <Train_Base_URL>/tspa/api/v1/network-name/networkregistry/{TrustSchemeName}
   Payload: Trust List Entries [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]
   Return Codes: [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]

3. GET - Fetch Trust List <Participating Service> individual TSP details

   Auth: Public
   Path: <Train_Base_URL>/tspa/api/v1/network-name/trustlist/tsp/individual/{TrustSchemeName}/{UID}
   Payload: TrustServiceProvidersDetails [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]
   Return Codes: [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]

4. GET - Fetch Trust List Participating Service TSP details

   Auth: Public
   Path: <Train_Base_URL>/tspa/api/v1/network-name/trustlist/tsp/individual/{TrustSchemeName}/{UID}
   Payload: TSP UID [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]
   Return Codes: [reference Swagger https://essif.iao.fraunhofer.de/swagger_undp_train/]

5. TRAIN API Swagger Endpoint [development]

   https://essif.iao.fraunhofer.de/swagger_undp_train/

6. Base Host Configurations

S.No	VM Type	Purpose	Hard Disk (GB)	RAM (GB)	Number of Machines	Software Required
1	Ubuntu 20 or later	Network Registry Host	64	4	1	Java 8, 17, Tomcat 9 or above, Nginx,
2	Ubuntu 20 or later	Provider Registry Host	64	4	1	Java 8, 17, Tomcat 9 or above, Nginx
3	Ubuntu 20 or later	Deploying Swagger and API endpoints	64	4	2	Java 8, 17, Tomcat 9 or above, Nginx
4	Debian	DNS Zone Manager	64	4	1	NSD Zone, Nginx
5	Debian	Secondary Servers	32	4	4	NSD Zone, Nginx

7. Template Trust Scheme (TSP) XML

Client Web App Configuration

The Client Web App has been developed using Node.js v18.9.0, KeyCloak v19.0.1 on JVM, and MongoDB 6.0.

Running the application on local (development) computer

1. **Prerequisites:**

   1.1. Selects a submission entry in a "Pending" state on the Submissions List page
   1.2. Install Node.js and npm
   1.3. Install MongoDB
   1.4. Install KeyCloak

2. **Step 1: Configure KeyCloak**

   2.1. Download the KeyCloak realm configuration template KeyCloak configuration
   2.2. Through the command line, open the KeyCloak installation folder.
   2.3. Start KeyCloak using the command "bin\kc.bat start-dev".
   2.4. Open your web browser and enter the URL http://localhost:8080 – this will bring you to the KeyCloak Administration Console.
   2.5. Click the "Administration Console" link, and follow the prompts to create a master account for this KeyCloak instance.
   2.6. Once signed in to your master account, click the dropdown with the "Master" option selected in the top left corner of the screen, and click the "Create Realm" button.
   2.7. Now, from the "Create Realm" page, use the realm configuration file downloaded earlier as the resource file.
   2.8. Set the realm name to "xxx_Realm"
   2.9. Click the "Create" button. The KeyCloak realm is now configured.
   2.10. Users for each user role will need to be created within the new realm.
   2.11. Leave the command line open while running the application.

3. **Step 2: Configure MongoDB**

   3.1. In the project, in the "config.json" file located under the "./data/MongoDB" directory, set the "PORT" property to match the port used by your MongoDB instance.
   3.2. Create a database entitled "xxx" in MongoDB

4. **Step 3: Run the Node.js application**

   4,1. Open a new command line window.
   4,2. Install the packages necessary to run the project by navigating to the project's root folder in the command line and running the command "npm i"
   4,3. After the packages are installed, run the application using the command "npm start".
   4,4. The local Node.js server should now be running.
   4,5. Open a web browser and enter the URL http://localhost:1337/ to view the running application.

Node.JS Packages

The following node.js packages were installed during the development of the application:

1. axios
2. body-parser
3. express
4. express-session
5. jsonwebtoken
6. mongoose
7. morgan
8. nodemon
9. openid-client
10. openid-client-helper
11. path
12. pug
13. serve-favicon

The packages used in the project (and their version numbers) are specified in the file "packages.json", which can be found in the root folder of the application.

TSPA and TSP Trust Scheme (Profile) Examples

Trust Schemes vary in the level of service detail and descriptive properties depending on the requirements of their publisher (TSPA) content. The example XML documents are Trust Schemes that define the Network Operator (performing TSPA and TPA roles), Intermediate Providers (performing TSPA and TPA roles) and Participating Services (performing a TSP role).

Trust Scheme Operator XML Properties Example

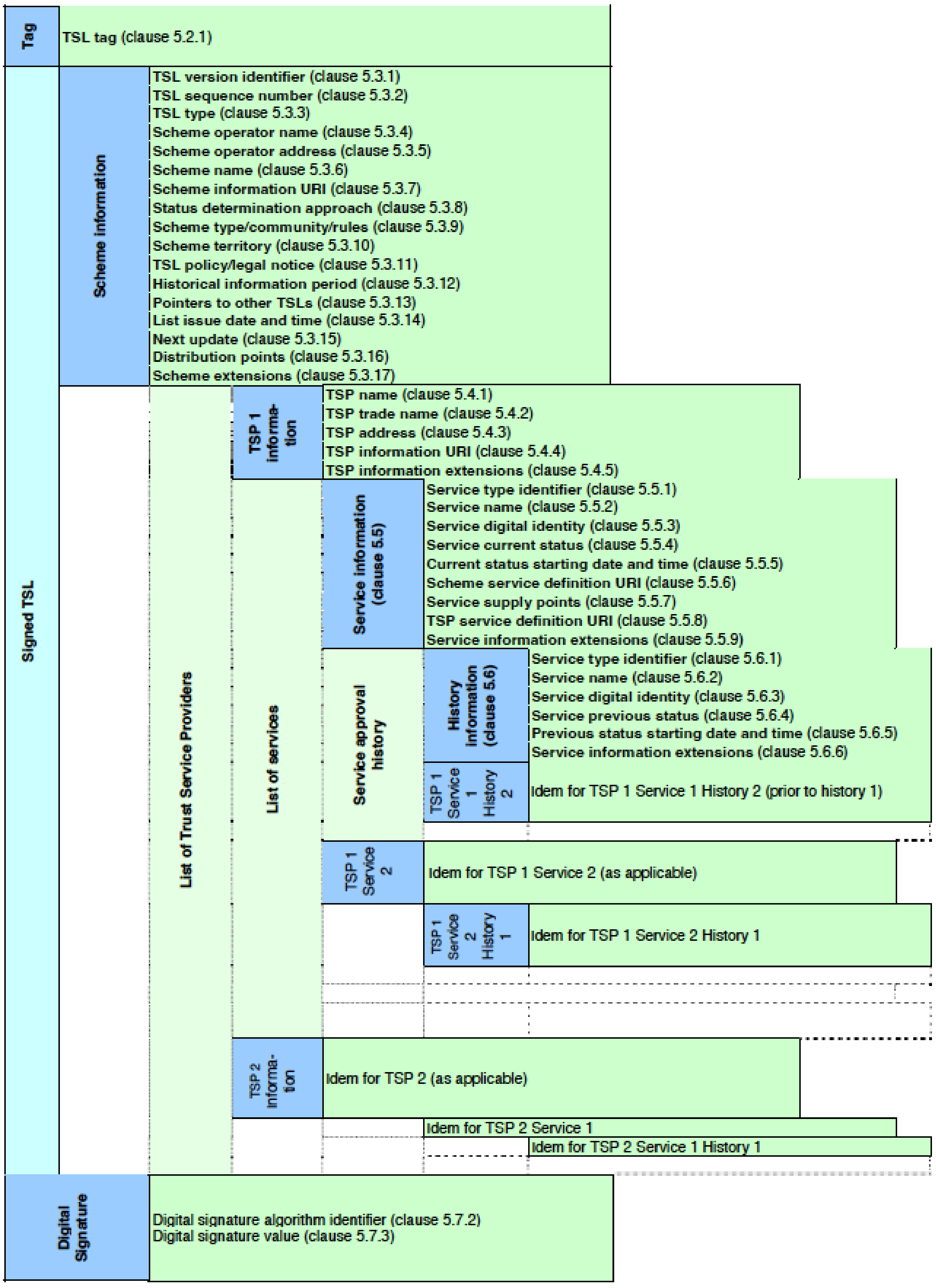
```
  …
 <SchemeInformation>
   <TSLVersionIdentifier>1</TSLVersionIdentifier>
   <TSLSequenceNumber>1</TSLSequenceNumber>
   <TSLType>https://TRAIN/TrstSvc/TrustedList/TSLType/networkname</TSLType>
   <SchemeOperatorName>
     <Name xml:lang="en">Network Operator</Name>
   </SchemeOperatorName>
   <SchemeOperatorAddress>
     <PostalAddresses>
        …
     </PostalAddresses>
     <ElectronicAddress>
       <URI xml:lang="en">mailto:contact.networkname.org</URI>
     </ElectronicAddress>
   </SchemeOperatorAddress>
   <SchemeName>
     <Name xml:lang="en">Trust Scheme of Network Operator</Name>
   </SchemeName>
   <SchemeInformationURI>
     <URI xml:lang="en">https://TRAIN/interoperability/Network-Registry</URI>
   </SchemeInformationURI>
   <SchemeTypeCommunityRules>
     <URI xml:lang="en">https://TrustScheme_TRAIN.example.com/en/network-registry-rules.html</URI>
   </SchemeTypeCommunityRules>
   <SchemeTerritory>Global</SchemeTerritory>
   <PolicyOrLegalNotice>
     <SchemeLegalNotice xml:lang="en">The applicable legal notice text to be displayed.</SchemeLegalNotice>
   </PolicyOrLegalNotice>
   <ListIssueDateTime>2021-12-15T00:00:00Z</ListIssueDateTime>
  </SchemeInformation>
  <TrustServiceProviderList>
```

Trust Registry Participant (TSP) XML Properties Example:

```
…
<TrustServiceProviderList>
    <TrustServiceProvider>
        <TSPInformation>
            <TSPName>
                <Name xml:lang="en">Health Service Provider Details</Name>
            </TSPName>
            <TSPRole xml:lang="en"> Issuer </TSPRole>
            <TSPLegalName>
                <Name xml:lang="en">Legal Name of Health Service Provider </Name>
            </TSPLegalName>
            <TSPTradeName>
                <Name xml:lang="en">Trade Name of Health Service Provider </Name>
            </TSPTradeName>
            <TSPEntityIdentifierList>
                <TSPEntityIdentifier>
                    <TSPEntityIdentifierName>
                        <Name xml:lang="en">Entity Identifier Details</Name>
                    </TSPEntityIdentifierName>
                    <TSPEntityIdentifierType xml:lang="en"> vLEI </TSPEntityIdentifierType>
                    <TSPEntityIdentifierValue xml:lang="en"> LEICode ISO 17442 </TSPEntityIdentifierValue>
                    <TSPEntityIdentifierURI xml:lang="en"> https://if_theres_a_link </TSPEntityIdentifierURI>
                </TSPEntityIdentifier>
            </TSPEntityIdentifierList>
            <TSPAddress>
                <PostalAddresses>
                    <PostalAddress xml:lang="en">
                    ...
                    </PostalAddress>
                </PostalAddresses>
                <ElectronicAddress>
                ...
                </ElectronicAddress>
            </TSPAddress>
    <TSPInformationURI>
      <URI xml:lang="en">https://www.government.nl/topics/coronavirus-covid-19</URI>
    </TSPInformationURI>
    <TSPAgentList>
      <TSPAgent>
        <TSPAgentName>
          <Name xml:lang="en">Agent Name 1</Name>
        </TSPAgentName>
        <TSPAgentAddress>
          <PostalAddresses>
            <PostalAddress xml:lang="en">
            ...
            </PostalAddress>
          </PostalAddresses>
          <ElectronicAddress>
          ...
          </ElectronicAddress>
        </TSPAgentAddress>
      </TSPAgent>
    </TSPAgentList>
    <TSPQualifierList>
      <TSPQualifier xml:lang="en">
      ...
      </TSPQualifier>
    </TSPQualifierList>
  </TSPInformation>
  <TSPServices>
    <TSPService>
      <ServiceInformation>
        <ServiceTypeIdentifier>https://train.trust-scheme.de/schema/networkname-schema.json</ServiceTypeIdentifier>
        <ServiceName>
          <Name xml:lang="en"> Services offered by Health Service Provider</Name>
        </ServiceName>
        <ServiceDigitalIdentity>
          <DigitalId>
            <DID>did:web:example.com:covid:registry</DID>
          </DigitalId>
        </ServiceDigitalIdentity>
        <ServiceStatus>http://networkregistrylookup.essif.trust-scheme.en/ServiceTypes/Servicestatus/active/</ServiceStatus>
        <StatusStartingTime>2021-12-15T00:00:00Z</StatusStartingTime>
        <SchemeServiceDefinition>https://github.com/minvws/nl-covid19-coronacheck-app-coordination</SchemeServiceDefinition>
        <ServiceSupplyPoint>https://www.npkd.nl/masterlist.html</ServiceSupplyPoint>
        <ServiceDefintionURI>https://www.example-textdesccriptionrequired</ServiceDefintionURI>
        <AdditionalServiceInformation>
          <ServiceCredentialTypes>
            <KeyType>DID</KeyType>
          </ServiceCredentialTypes>
          <ServiceGovernanceURI>https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32021R0953</ServiceGovernanceURI>
        </AdditionalServiceInformation>
      </ServiceInformation>
    </TSPService>
  </TSPServices>
</TrustServiceProvider>
```

ETSI Logical Model

ETSI Standard document reference: Logical Model for the ETSI 119 612 TS


Tag
TSL tag (clause 5.2.1)
Signed TSL
Scheme information
TSL version identifier (clause 5.3.1)
TSL sequence number (clause 5.3.2)
TSL type (clause 5.3.3)
Scheme operator name (clause 5.3.4)
Scheme operator address (clause 5.3.5)
Scheme name (clause 5.3.6)
Scheme information URI (clause 5.3.7)
Status determination approach (clause 5.3.8)
Scheme type/community/rules (clause 5.3.9)
Scheme territory (clause 5.3.10)
TSL policy/legal notice (clause 5.3.11)
Historical information period (clause 5.3.12)
Pointers to other TSLs (clause 5.3.13)
List issue date and time (clause 5.3.14)
Next update (clause 5.3.15)
Distribution points (clause 5.3.16)
Scheme extensions (clause 5.3.17)
List of Trust Service Providers
TSP 1 informa-tion
TSP name (clause 5.4.1)
TSP trade name (clause 5.4.2)
TSP address (clause 5.4.3)
TSP information URI (clause 5.4.4)
TSP information extensions (clause 5.4.5)
List of services
Service information (clause 5.5)
Service type identifier (clause 5.5.1)
Service name (clause 5.5.2)
Service digital identity (clause 5.5.3)
Service current status (clause 5.5.4)
Current status starting date and time (clause 5.5.5)
Scheme service definition URI (clause 5.5.6)
Service supply points (clause 5.5.7)
TSP service definition URI (clause 5.5.8)
Service information extensions (clause 5.5.9)
Service approval history
History information (clause 5.6)
Service type identifier (clause 5.6.1)
Service name (clause 5.6.2)
Service digital identity (clause 5.6.3)
Service previous status (clause 5.6.4)
Previous status starting date and time (clause 5.6.5)
Service information extensions (clause 5.6.6)
TSP 1 Service 1 History 2
Idem for TSP 1 Service 1 History 2 (prior to history 1)
TSP 1 Service 2
Idem for TSP 1 Service 2 (as applicable)
TSP 1 Service 2 History 1
Idem for TSP 1 Service 2 History 1
TSP 2 informa-tion
Idem for TSP 2 (as applicable)
Idem for TSP 2 Service 1
Idem for TSP 2 Service 1 History 1
Digital Signature
Digital signature algorithm identifier (clause 5.7.2)
Digital signature value (clause 5.7.3)

References

1. TRAIN project summary: https://gitlab.grnet.gr/essif-lab/infrastructure/fraunhofer/train_project_summary
2. ETSI v2.2.1 https://www.etsi.org/deliver/etsi_ts/119600_119699/119612/02.02.01_60/ts_119612v020201p.pdf