Digital Consent Architecture. Estonia and Finland

Andres Kütt

September 27, 2018

The document summarises the key ideas of privacy-preserving data sharing between public and private sector institutions. They are set in the context of consensual sharing of patient information in Estonian e-health ecosystem. The solution draws heavily on the Finnish MyData concept but diverges in terms of the proposed technical solution.

License

Background

Although in many cases[1] the legal framework in place provides a solid foundation to privacy-respecting data[2] access, a blanket consent is not very granular. By definition, such blanket consent is put in place where there is reason to believe it is in the best interest of the overwhelming majority of the citizens. Where such a reason does not exist[3], blanket consent typically does not exist and data is not shared. This is often not in the interest of neither the citizen nor the organisations accessing or hosting the data.

Moreover, a blanket consent can be controversial in nature as it forces a tradeoff between the interests of a small majority and a vast majority and the decision to provide it is inherently subjective.

Finally, the blanket consent is by design slow to respond to the changing environment as it should not be granted easily.

Therefore, a mechanism needs to be put in place that allows for granular consenting of data access or denial thereof[4] by the data proprietor.

Description of that mechanism constitutes the core of this document.

The idea

The basic idea here is to give citizens *control* over the data organisations hold about them[5]. This is explicitly different from the idea of giving citizens their *data*. The difference stems from the following assumptions:

[1] In both Estonian and Finnish legal context agencies can have a blanket permission to access data if they are permitted to ask the same data from the customer

[2] Hereinafter "data" and "APIs that give access to data or services" are used synonymously

[3] E.g. in most cases where private sector would be accessing data held by the public sector; areas that are restrictive by default like genetics and healthcare; areas covered by specific data protection measures like finance and tax etc.

[4] An "negative consent" is simply a reverse of what a consent is. Instead of all data being inaccessible unless a consent is in place, all data is accessible unless access is explicitly denied

[5] This assumes data proprietors share a citizen identity: otherwise it would be impossible to map a person to the data being held

- The regular need for citizens to download their data reduces data fidelity (e.g. if a user downloads their electricity meter reading once a day, they won't get intra-day movements)
- The burden of having to both retain and protect their data is something an average citizen can handle worse than an organisation legally obliged to do so
- Increasing the number of copies[6] of a data point exponentially increases the chance of a breach

The solution

The basic idea is to build a multi-centred[7] trust authority that maintains a list of active authorisations[8] for data access, each digitally signed by the citizen.

A common service flow would look like so:

1. The user expresses the desire for an app to access their private data or service hosted by the data proprietor[9]
2. The app sends a user to a trust service of their choice
3. At the trust service, the user signs an authorisation for the app to access their data
4. The user is sent back to the app with a token identifying the act of authorisation
5. The app shows up at the data proprietor with a request for data or service and a token
6. The data proprietor validates the token against the trust service and makes sure the authorisation matches the data requested
7. The data proprietor shares the data with the app

Figure 1 illustrates the interaction in detail.

Key considerations

The solution raises the following considerations

- Albeit only sharing data for immediate consumption, the data consumer can store data for unauthorised processing[10]
- Since the citizen authorises the transaction, they must take the full responsibility of assuring the organisation authorised has the necessary means of protecting the data shared with them

[6] If a data point is to be re-used by another organisation, three copies of it would necessarily be created for the download scenario: the original source of the data, the copy of the citizen and the copy of the organisation using it

[7] A number of service providers implementing standardised APIs

[8] A blockchain would require less centralised trust than a database but would reduce the privacy guarantees as authorisations would essentially be public. The fact of authorisation itself, however, can be considered private information

[9] To differentiate from the data owner, the citizen, the organisation responsible for collecting and securing the data is called an "proprietor"

[10] Contrary to the download model, two copies are created (the proprietor and the app) instead of three. Also, the copy is created not on a regular basis but on demand

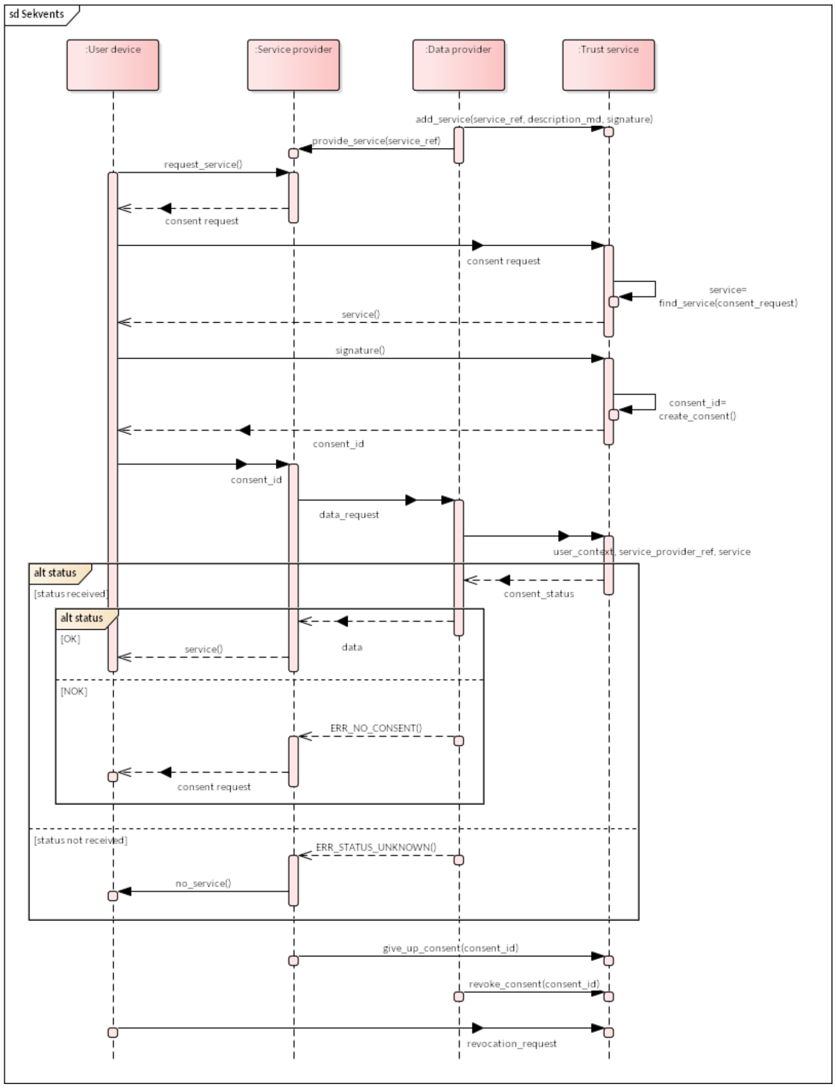

Figure 1: A technical sequence diagram of the authorisation process

- A central point of trust is created that can be breached. However
  - The data is not immediately lost as it still resides with the proprietor and data access can be protected by additional technical and legal means (i.e. breaching both the trust server *and* an organisation the data is shared with becomes necessary)
  - The authorisations can be secured via a personal digital signature of the citizen

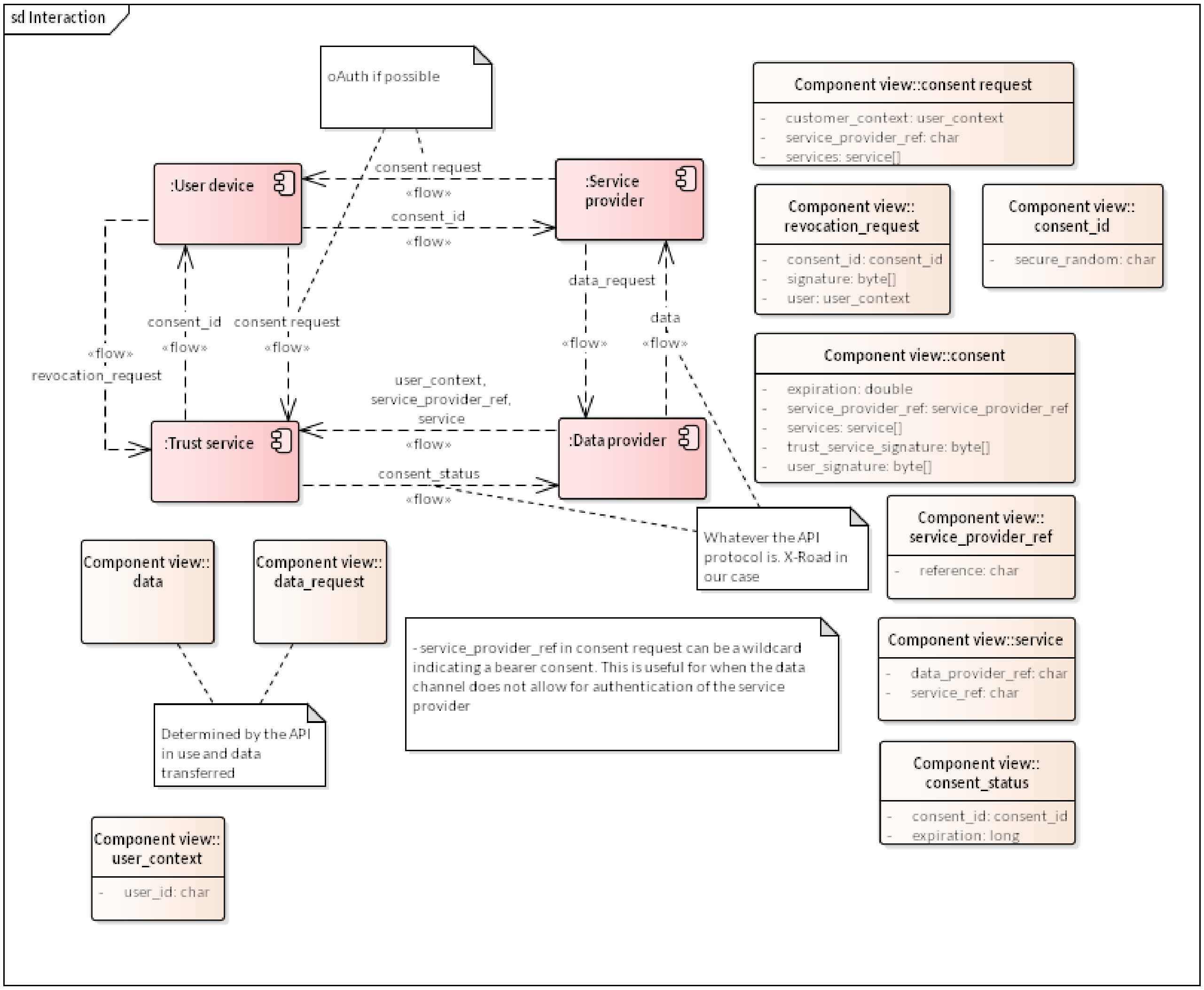
sd Interaction
oAuth if possible
:User device
:Service provider
:Trust service
:Data provider
consent request
«flow»
consent_id
«flow»
data_request
«flow»
data
«flow»
consent_id
«flow»
consent request
«flow»
«flow»
revocation_request
user_context, service_provider_ref, service
«flow»
consent_status
«flow»
Whatever the API protocol is. X-Road in our case
Component view::consent request
- customer_context: user_context
- service_provider_ref: char
- services: service[]
Component view:: revocation_request
- consent_id: consent_id
- signature: byte[]
- user: user_context
Component view:: consent_id
- secure_random: char
Component view::consent
- expiration: double
- service_provider_ref: service_provider_ref
- services: service[]
- trust_service_signature: byte[]
- user_signature: byte[]
Component view:: service_provider_ref
- reference: char
Component view::service
- data_provider_ref: char
- service_ref: char
Component view:: consent_status
- consent_id: consent_id
- expiration: long
Component view:: data
Component view:: data_request
Determined by the API in use and data transferred
- service_provider_ref in consent request can be a wildcard indicating a bearer consent. This is useful for when the data channel does not allow for authentication of the service provider
Component view:: user_context
- user_id: char

Figure 2: A detailed technical interaction diagram of the authorisation process

Implementation

An implementation of the solution does not exist as of September 27, 2018. However, there is an agreement between a number of parties (including Estonian e-health data proprietor, technology providers, a startup and the IT arm of Estonian Ministry of Social Affairs, etc.) on a technical system design that would implement such a flow.

The current proposed solution assumes an underlying transport layer with the following capabilities

- End to end encryption of all traffic (i.e. no third party can access the data shared)
- Authentication and explicit access rights to the data sharing APIs (i.e. data is only shared with known organisations whose identity matches the one on the authorisation artefact)
- Non-repudiation and audit logging (i.e. neither data proprietor, the trust service nor the service provider can claim a particular interaction did not take place)

A number of ways to build such a transport layer can be devised, the proposed solution uses X-Road[11] simply because of its ready availability and low friction in the given context.

Figure 2 contains the proposed implementation along with a rudimentary data model. The following business processes are missing from the model:

- Establishment of the service list. This should be agreed upon between the trust service, service provider and the data provider[12] as each plays a role:
  - The trust service, as a neutral third party, presents the user with the full legal explanation of what the service entails based on input from the data provider
  - The data provider must implement a technical API that matches the description presented by the trust service
  - The service provider must consume the API
- Audit, billing etc. that are based on the consent ID passed around for this explicit purpose[13] and the underlying transport layer
- Failure of trust by
  - The trust service[14], at which point all data providers *must* stop accepting consents from that provider
  - The service provider, at which point the service provider *must* loose access to the transport layer[15]

[11] See `https://www.niis.org/data-exchange-layer-x-road/`

[12] And published in a machine-readabale standardised format to ease interoperability

[13] It can be dropped from all interactions without loss of immediate functionality

[14] This implies existence of a supervision authority

[15] This implies a mechanism to monitor service provider behaviour and a readiness by the transport layer operator to accept external requests for measures like identity revocation