PDaaS - Personal Data as a Service

lucendio

November 2, 2016

Abstract

TODO

Contents

Chapter 1

Introduction

1.1 Abstract

1.2 List of Terms

Chapter 2

Overview

2.1 Features

2.2 Roles

2.3 Usecases

2.4 Architecture

2.5 Components

2.6 Dependencies

2.6.1 Depending on following standards:

- JSON Object Signing and Encryption

2.6.2 Depending on following technologies

Chapter 3

Data

- What data the user should be able to manage or to control?
- request should contain a specific value for how precise a certain data should
  be (e.g. location)
- differentiate between data the user has and collects about herself, and data others have/create
  on the user (e.g. doctors/e-health, banks, public authorities) - depending on that, how does
  the data get into the system?
- data precision/resolution? (maybe it might be legit to say: the more precise the data is, the
  higher the (selling) value)
- when responding with data, the contents should be labeled with an expire date, set by the
  owner of the data
- NOTE, in reality, certain business processes might require to store some data, e.g. for legal
  a/o administrative reasons - address to reproduce where amazons package went

3.1 Data Categories

NOTE: does not mean data types!

- profile data
- sensor data (e.g. geo-location, motions)
- financial record (and history)
- payment information
- medical record
- governmental data
- biometric datasets (e.g. finger print, retina)
- web search history
- what about user-created content, like pictures, videos or blog posts?

[A-middleware-architecture-for-privacy-protection_2007 p4r] suggests:

- active
- semi-active
- passive

3.2 Data types

- String (number, boolean) - single values, which can go into the db
- serialized data structure (as string); requires schema upfront
- data series based on time (recurring data with familiar structure and purpose)
- files - text or binary - need to get written to a filesystem and referenced by link with a
  db entry OR gets stored directly into the db

3.3 Dev API

- devs can registering new data types by providing a type name, description and a schema
- querying data by using graphql would provide the ability to ask fine-granted and structured
  based on a schema-like way
- every schema needs a version; we need to store every schema version

Chapter 4

Concepts

4.0.1 remote computation (aka block-chain, MITs SafeAnswer)

- during execution, http requests could be allowed, if information, such as domain, protocol,
  request and response body, mime-type will be provided up front (like *Content Security Policy*)
- we might need some kind of formal assurance policy (aka contract) signed by both sides

Chapter 5

Architecture

What is this data service look like? Which parts and tasks does it cover?

- hard-connection backend and mobile device? (for continuously collection data e.g.
  geo-location) –> internal aggregation;
  or soft-connection between them –> just exposing api to store certain types of arbitrary data,
  collected and stored/cached on external sources
- automated backup mechanisms, e.g. distribute encrypted container, maybe on a regular basis,
  to dropbox (or the like) or git repository or maybe scp
- 2-factor auth? How?
- an inclusion of a mobile device (as the actual data vault/container) could make using
  2-factor-auth more obvious and easier to implement - we might get in for free depending on
  we model the interaction processes
- data backup?
- explicitly no ecosystem approach: no interoperability between each each individuals PDS,
  thus no market/appstore as business model
- either cloud-based (not in the near of the user) and/or local (users device) storage

Chapter 6

API

Chapter 7

Interfaces

How third parties can interact with such data sources?

- when requesting time series data, which concepts should been supported:
  A) collection data within the service and only after a period of time handing over the data
  (after filtering them)
  B) providing some sort of real time feed through the service (filtering upfront)?
- after introducing the system and because it is likely that vendors will support this way of
  exchanging/acquiring data only if its feasible and enough user are already using it. So to
  increase adoption rate of users, it might be supportive to provide a browser extension to
  mimic the role of the vendor somehow (like *Personal* did before
  [https://techcrunch.com/2011/11/17/personal-is-a-secure-vault-for-all-of-your-private-digital-data/]);
  also it would be great to have some client-plugins for several platforms, languages and
  frameworks right from the start (e.g. nodejs, shopify, rails, wordpress etc.)
- spam protection? if we have a general URI (endpoint) for incoming requests, which had to
  get submitted upfront, that identifier (URI) might get known com-

monly, thus unintended mass
requests might occur. possible solution: generate URI+token (token = cert fingerprint?)
within the software? (how do we get it from there to the third-party?)
OR let the user
create a human readable slug on the fly during the external interaction - therefor on the
mobile notification we have at least sth to validate, but with no tech. assistance, and
it is still not spam-safe?

Chapter 8

Components

8.1 mobile app

- notification receiver
- permission manager
- admin panel
- data collector (geo-location, movement, health data, etc.)

8.2 statistics, monitoring, self-healthiness and logging

- logging request of external services (when, what, why)
- watching and detecting anomalies (e.g. cert changed, domain changed, etc.)

8.3 Core

- self-detection host system to measure how secure the env is, I am currently in

Chapter 9

Security

9.0.1 providing self-collected (meta) data

- when handing over large amount of meta data (e.g. geo location log over time), these data have
  to be anonymized before leaving the system (see: MITs *openPDS* project, more precisely
  SafeAnswer)
- there have to be a protocol for such a process (details on what data they exactly need)
- end-to-end encryption everywhere, or when its sufficient transport encryption only?

Chapter 10

Visualization

- what should the admin panel look like?
- How should the data be visualized - and where?

Chapter 11

Interactions

- what type of interaction patters are available - and where?