Tutorial Outline

PART I. Personal Data Management Systems (PDMS)
Review of functionalities & addressed privacy threats
Individual’s PDMS vs (corporate) DBMS and main properties to achieve

PART II. TEE-based Data Management
The promises of Trusted Execution Environments (TEEs)
A review of privacy-preserving data management using TEEs

PART III. Bridging the Gap between PDMS and TEEs
How could the main properties be achieved?
A quick view of remaining challenges
10 years history of Personal Data Management Systems

Since 2008 – FreedomBox@Columbia (Eben Moglen)
Free individuals from state control
PDMS = Low-cost open HW + open SW

Since 2010 – PDS@Inria [AAB+10], MiloDB [ABP+14], PDMS [ABB+19]
Manage (specific) personal folders at hand, enforce privacy policies
PDMS = Tamper resistant HW (smart card or TEEs) + embedded DBMS

2012 – OpenPDS@MIT [MSW+14], 2016 – DataBox-BBCBox@Nottingham [MZC+16]
Manage your data locally, externalize only safe answers
PDMS = SW running on user’s device (smartphone, tablet)

Since 2013 – Gov. [MyDex, MesInfos] & commercial initiatives [NextCloud, Cozy, …]
Collect personal data from different data silos & provide transversal Apps
PDMS = Online SW with Apps (terminology shift: PDS → personal cloud)

Since 2018 – Solid PODs and Inrupt (Tim Berner Lee)
To re-decentralize the Web of personal data, give agency to individuals
PDMS = Personal Online Data store (PODs)
10 years history of Personal Data Management Systems

Since 2008 – FreedomBox@Columbia (Eben Moglen)
Free individuals from state control
PDMS = Low-cost open HW + open SW

Since 2010 – PDS@Inria [AAB+10], MiloDB [ABP+14], PDMS [ABB+19]
Manage (specific) personal folders at hand, enforce privacy policies
PDMS = Tamper resistant HW (smart card or TEEs) + embl

2012 – OpenPDS@MIT [MSW+14], 2016 – DataBox@Bergen
Manage your data locally, externalize only safe answers
PDMS = SW running on user's device (smartphone, tablet)

Collect personal data from different data silos & provide transversal Apps (terminology shift: PDS → personal cloud)

Since 2018 – Solid PODs and Inrupt (Tim Berner Lee)
Tow: decentralized the Web of personal data, give agency to individuals
PDMS = Personal Online Data store (PODs)

What are their functionalities?
What are the privacy threats considered?
Main classes of architectures for a PDMS

Online personal cloud

E.g., Cozy, Digi.me, NextCloud, BitsAbout.Me, Perkeep

Functionality:
- Data collectors for everything (banks, energy, health, geolocation, ‘likes’ graphs, ...)
- Personal (cross-)computation (1 individual) features for App developers
- Backup (full retention: Perkeep)

Trust model:
- Personal cloud provider & Apps considered fully honest
- Security standards, PEN tests (Cozy), code transparency (community checks)

No-knowledge personal cloud

E.g., MyDex, SpiderOak, Digi.me

Functionality:
- Secure data store, personal data encrypted (encryption keys managed at client side)
- Secure backup and point in time recovery

Trust model:
- Personal cloud provider is untrusted (but the client device is not)
- Considered attacks: data snooping and secondary usages (server), ransomware (client)
Main classes of architectures for a PDMS

Online personal cloud ➔ Advanced functionality, strong trust assumptions

- E.g., Cozy, Digi.me, NextCloud, BitsAbout.Me, Perkeep
- Functionality:
  - Data collectors for everything (banks, energy, health, geolocation, ‘likes’ graphs, …)
  - Personal (cross-)computation (1 individual) features for App developers
  - Backup (full retention: Perkeep)
- Trust model:
  - Personal cloud provider & Apps considered fully honest
  - Security standards, PEN tests (Cozy), code transparency (community checks)

No-knowledge personal cloud ➔ Increased security, minimalist functionality

- E.g., MyDex, SpiderOak, Digi.me
- Functionality:
  - Secure data store, personal data encrypted (encryption keys managed at client side)
  - Secure backup and point in time recovery
- Trust model:
  - Personal cloud provider is untrusted (but the client device is not)
  - Considered attacks: data snooping and secondary usages (server), ransomware (client)
Main classes of architectures for a PDMS (cont.)

Home (or edge) cloud software

E.g., OpenPDS [MSW+14], Databox [MZC+16]

Functionality:
- Trusted storage on end-user device or at the edge (1 store per IoT device)
- Personal computation provided safe answers and aggregated views, never raw data
- Data dissemination rules to share computed results

Trust model: user device and SW must be trusted

Home cloud plugs (dedicated)

E.g., FreedomBox, Cloud Locker

Functionality: data store and backup in a dedicated hardware plug

Trust model: Plug code must be trusted (dedicated => limited attack surface)

Tamper-resistant home cloud

E.g., PDS [AAB+10], PlugDB [ANSP14, ALSP+15, LASP+17, ABB+19]

Functionality: (simple) store, share, compute (local/global) in a secure HW device

Trust model: secure HW + embedded SW are trusted
Main classes of architectures for a PDMS (cont.)

Home (or edge) cloud software ➔ ‘formal’ security lost, more functionality

E.g., OpenPDS [MSW+14], Databox [MZC+16]

Functionality:
- Trusted storage on end-user device or at the edge (1 store per IoT device)
- Personal computation provided safe answers and aggregated views, never raw data
- Data dissemination rules to share computed results

Trust model: user device and SW must be trusted

Home cloud plugs (dedicated)

E.g., FreedomBox, CloudLocker

Functionality: data store and backup

Trust model: Plug code must be trusted ➔ Security at the price of functionality, advanced processing on untrusted device

Tamper-resistant home cloud

E.g., PDS [AAB+10], PlugDB [ANSP14, ALSP+15, LASP+17, ABB+19]

Functionality: (simple) store, share, compute (local/global) in a secure HW device

Trust model: secure HW + embedded SW are trusted
## Synthesis : functionalities

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage</strong></td>
<td>Online personal cloud</td>
</tr>
<tr>
<td></td>
<td>Regular DBMS</td>
</tr>
<tr>
<td><strong>Backup</strong></td>
<td>Regular DBMS</td>
</tr>
<tr>
<td><strong>Data collection</strong></td>
<td>Web scrapping</td>
</tr>
<tr>
<td><strong>Personal computations</strong></td>
<td>Linked/ transversal queries</td>
</tr>
<tr>
<td><strong>Distributed computations</strong></td>
<td>[synchro.]</td>
</tr>
<tr>
<td><strong>Data dissemination</strong></td>
<td>[synchro.]</td>
</tr>
</tbody>
</table>

1- The whole personal cloud data life-cycle must be covered!
### Synthesis: functionalities (cont.)

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Online personal cloud</strong></td>
</tr>
<tr>
<td>Storage</td>
<td>Regular DBMS</td>
</tr>
<tr>
<td>Backup</td>
<td>Regular DBMS</td>
</tr>
<tr>
<td>Data collection</td>
<td>Web scrapping</td>
</tr>
<tr>
<td>Personal computations</td>
<td>Linked/ transversal queries</td>
</tr>
<tr>
<td>Distributed computations</td>
<td>X</td>
</tr>
<tr>
<td>Data dissemination</td>
<td>[synchro.]</td>
</tr>
</tbody>
</table>

1- The whole personal cloud data life-cycle must be covered!
2- Distributed computations are poorly covered…
Less useful? No, Big-Data perspectives!
More difficult? Yes, efficient and secure (solutions in the tamper resistant context)
### Synthesis: trust

#### Representative Personal Cloud approaches

<table>
<thead>
<tr>
<th>Considered threats</th>
<th>Online personal cloud</th>
<th>Zero-knowledge personal cloud</th>
<th>Home cloud software</th>
<th>Home cloud Plug</th>
<th>Tamper resistant personal server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data snooping</td>
<td></td>
<td>Data leakage</td>
<td>Data snooping</td>
<td>Data snooping</td>
<td>Data snooping</td>
</tr>
<tr>
<td>Data leakage</td>
<td></td>
<td>2\textsuperscript{nd} usages</td>
<td>Data leakage</td>
<td>Data leakage</td>
<td>Data leakage</td>
</tr>
<tr>
<td>2\textsuperscript{nd} usages</td>
<td></td>
<td>Client Failure</td>
<td>2\textsuperscript{nd} usages</td>
<td>2\textsuperscript{nd} usages</td>
<td>2\textsuperscript{nd} usages</td>
</tr>
</tbody>
</table>

#### Trust model

- **Fully-honest personal cloud & Apps**
- **Semi-honest or Malicious personal cloud**
  - Trusted Apps
  - Trusted client
- **Trusted personal cloud**
  - Trusted Plug
  - Trusted Apps
- **Trusted personal cloud**
  - Semi-honest infra.
  - Untrusted Apps

#### Privacy and security measures

- **Security stds, Business model**
- **Open source**
- **Client-side encrypt°**
  - ‘no-knowledge’ store
- **Safe answers**
- **Separated stores**
- **Local audit**
- **Closed platform**
  - (dedicated device), physical ownership
- **Secure HW**
  - small TCB
  - Secure distributed protocols

---

1. different privacy threats considered, all must be circumvented
### Synthesis: trust (cont.)

<table>
<thead>
<tr>
<th>Considered threats</th>
<th>Online personal cloud</th>
<th>Zero-knowledge personal cloud</th>
<th>Home cloud software</th>
<th>Home cloud Plug</th>
<th>Tamper resistant personal server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data snooping</td>
<td>Data snooping</td>
<td>Data snooping</td>
<td>Data snooping</td>
<td>Data snooping</td>
<td>Data snooping</td>
</tr>
<tr>
<td>Data leakage</td>
<td>Data leakage</td>
<td>Data leakage</td>
<td>Data leakage</td>
<td>Data leakage</td>
<td>Data leakage</td>
</tr>
<tr>
<td>2ndary usages</td>
<td>2ndary usages</td>
<td>2ndary usages</td>
<td>2ndary usages</td>
<td>2ndary usages</td>
<td>2ndary usages</td>
</tr>
<tr>
<td>Client Failure</td>
<td>Client Failure</td>
<td>Client Failure</td>
<td>Client Failure</td>
<td>Client Failure</td>
<td>Client Failure</td>
</tr>
<tr>
<td>Ransomware</td>
<td>Ransomware</td>
<td>Ransomware</td>
<td>Ransomware</td>
<td>Ransomware</td>
<td>Ransomware</td>
</tr>
</tbody>
</table>

#### Trust model

- **Fully-honest personal cloud & Apps**
- **Semi-honest or Malicious personal cloud**
  - Trusted Apps
  - Trusted client
- **Trusted personal cloud**
  - Trusted client
  - Untrusted Apps
- **Trusted personal cloud**
  - Trusted Plug
  - Trusted Apps
- **Trusted personal cloud**
  - Semi-honest infra.
  - Untrusted Apps

#### Privacy and security measures

- **Security stds, Business model**
- **Open source**
- **client-side encrypto**
  - ‘no-knowledge’ store
- **Safe answers**
  - Separated stores
  - Local audit
- **Closed platform**
  - (dedicated device), physical ownership
- **Secure HW**
  - small TCB
  - secure distributed protocols

1- **different privacy threats considered, all must be circumvented**

2- **unifying the solutions is not trivial (if not impossible)**

Wide spectrum of architectural choices…

… but different – irreconcilable – trust models and security measures
Personal Data Management: anything new?

Objective:
(1) provide the set of functionalities
(2) address all threats

Decades of research in
..... secure data collection, storage, backup, queries!

Next:
Specificities of (individual’s) PDMS vs (corporate) DBMS
..... and derived properties for an extensive and secure PDMS
In [ABB+19]: 5 properties are defined…
Expected PDMS functionalities & properties: Data Collection

Corporate DBMS
A basic operation using wrappers/APIs
Well-known & predefined wrappers/APIs
… audited and patched by the admins

Individual’s PDMS
Primary data directly fed into user’s PDMS
Secondary data needs data scrapping
Huge set of scrappers
… with untrusted code (e.g., Weboob)
… accessing sensitive data (credentials)
… in an untrusted environment!

Property: A PDMS enforces piped data collection iff:
1- the only PDMS data, accessible to the data collector, is the credentials;
2- the credentials/collected data cannot be leaked outside the PDMS.
   The only external channel provided to the data collector is with a single data provider
   … and the code is suitably isolated not to leak data elsewhere
Expected PDMS functions & properties:  

Corporate DBMS  
- Computations on corporate data  
  - Set of (trusted) applications selected, … audited and patched by admins

Individual’s PDMS  
- Apps crossing several data from individual  
  - For the PDMS owner or an external service (e.g., Pay as you drive).
  - Apps ‘move’ to data but…
  - Apps are untrusted (user’s viewpoint)
    - → local data must not leak
  - Computations are untrusted (service viewpt)
    - → results must be attested

Property: A PDMS enforces **bilaterally trusted computations** iff:
1- the data computation can only access the expected data from the PDMS;
2- only the final result – not the raw data – can be exposed to a 3rd party;
3- it provides a proof that the result was produced by the expected code.
  ‘Bilateral’ → guarantees to the owner and the 3rd party willing to execute code
  To owner: minimal collection principle is fulfilled, raw data cannot leak
  To 3rd party: code remotely sent has been computed (it may include any verification on data)
Expected PDMS functions & properties: Collective computations

Corporate DBMS
Not common → practical solutions
e.g., few Hospitals run a collective query
A trusted party may be used (by contract)
SMC usable [BEE+17] (few participants)

Individual’s PDMS
Common → new solutions are needed
e.g., Big Data and IA (recommendations, participative studies, community learning…)
Mutual confidentiality & integrity are critical
At a very large scale
(no trusted party nor SMC)

Property: A PDMS enforces *mutually trusted collective computations* iff:
1- the data computation can only access the required participant data;
2- only the final result – not the raw data – can be exposed to a 3rd party or any participant;
3- it provides a proof that the result was produced by the expected code on the expected set of participants.
‘Mutual’ → guarantees also hold between the participants
Definition of an Extensive and Secure PDMS (ES-PDMS)

An Extensive & Secure PDMS

- provides the expected set of functionalities to cover the complete data life-cycle
  - data collection,
  - storage and recovery,
  - cross-computations,
  - collective computations,
  - data dissemination.

- and is compliant with their respective security properties counterparts,
  - piped data collection,
  - mutual data at rest protection,
  - bilaterally trusted personal computation,
  - mutually trusted collective computation,
  - controlled data dissemination.

How do we get there?

The field of TEE-based secure data management is rapidly developing
⇒ let’s take a closer look…
Thanks!

Questions?
References (1)


References (2)


References (3)


References (4)


