A Management Framework for Secure Multiparty Computation in Dynamic Environments

Marcel von Maltitz, Stefan Smarzly, Holger Kinkelin, Georg Carle
vonmaltitz@net.in.tum.de

Technical University of Munich (TUM)
Department of Informatics
Chair of Network Architectures and Services

Taipei, 23.04.2018
Outline

1. Motivation: Data Processing in Smart Environments
2. Problems for Privacy
3. Background: Secure Multiparty Computation
4. Migration to SMC
5. Technical Overview
Motivation

Smart Environments are equipped with a variety of sensors in each room.
A common use case is providing aggregated sensor and user data to:

- support automatic controllers (e.g. HVAC and lighting)
- enable interaction interfaces (e.g. voting-based room configuration)
- inform users (e.g. public displays)
Problems

Central Infrastructure = Trusted Third Party

Conflicts with Privacy Requirements:

- Raw data accessible by TTP
- Data usage intransparent
- Revocation of data
Modelling

- Data gathering initially decentralized
- Data owner ≠ data processor ≠ data consumer
- Data usage: Aggregated local values for remote consumer
- Individual data more critical than aggregates
- Privacy [1-6] means
  - Data minimization
  - Unlinkability / Purpose binding
  - Transparency / Usage insights
  - Intervenability / Control over own data
Background: Secure Multi-Party Computation

Definition (cf. [7]):
There are $n$ parties $P_1, \ldots, P_n$. Each party $P_i$ holds a secret value $x_i$.

Secure Computation of $y = f(x_1, \ldots, x_n)$ is performed if two conditions are satisfied:
- Correctness: the correct value of $y$ is computed
- Privacy: $y$ is the only new information that is released

Example: Addition

<table>
<thead>
<tr>
<th>Party</th>
<th>$x_i$</th>
<th>Share $P_1$</th>
<th>Share $P_2$</th>
<th>Share $P_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>$P_2$</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>$P_3$</td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Result</td>
<td>22</td>
<td>8</td>
<td>5</td>
<td>9</td>
</tr>
</tbody>
</table>
From TTP to SMC: Challenges

Dynamic Environment
- Parties previously unknown
- Subsets of Parties
- Different input data
- Computations previously unknown

Orchestration of Computations
- Synchronized communication
- No error handling

Service character
- Access for data consumers
- Metadata about available information
- Only parties obtain result
Architectural Overview: Hybrid Approach

Virtual Centrality

- Introduction of gateway (GW) for SMC network
- Single, generic endpoint for requests
- Hides complexity and fragility of SMC network

Decentralization

- Self-management
- Local storage of raw values
- Only reveal processed data via collaborative computations (SMC)
Peers | Self-Organization

Operation layer: execution of job phases

- Zeroconf
- Find Gateway
- Match capabilities
- X.509 certs exchange
- (Out-of-bound verification)
- Request verification
- Prepare request for computation
- Connect peers with each other
- Check for problems
- Synchronized trigger of session(s)
- Stream results to requesting client
Gateway
Realized Features

**Applicability**
- Adaptiveness in dynamic environments
- Automatic session configuration
- Automated and continuous execution of SMC
- Robustness of computations

**Privacy**
- Confidentiality
- Unlinkability of data
- Data minimization
- Transparency of data-processing
- Intervenability for peers

---

SMC Service Clients
- Gateway
- Peers

A Management Framework for SMC in Dynamic Environments | Marcel von Maltitz
Conclusion

- Secure multiparty computation realizes/supports realization of privacy properties
- New challenges arise when applying SMC in dynamic contexts
- We propose a wrapper around SMC to solve to these problems
- Then, SMC can be used as a robust service for continuous and automated computations