Optimized and Controlled Provisioning of Encrypted Outsourced Data

Anis Bkakria\textsuperscript{1} Andreas Schaad\textsuperscript{2} Florian Kerschbaum\textsuperscript{2} Nora Cuppens-Boulahia\textsuperscript{1} David Gross-Amblard\textsuperscript{3}

\textsuperscript{1}Télécom Bretagne, Rennes, France
\textsuperscript{2}SAP AG, Karlsruhe, Germany
\textsuperscript{3}Université de Rennes 1, Rennes, France

\textit{SACMAT 2014, London, ON, Canada.}
Contents

1 Introduction
   - Context and Goal
   - Problem
   - Our Contribution

2 Policy Configuration
   - System modeling
   - Policy modeling
   - Policy conflict detection
   - Policy satisfaction

3 Conclusion and future work
Contents

1 Introduction
   - Context and Goal
   - Problem
   - Our Contribution

2 Policy Configuration
   - System modeling
   - Policy modeling
   - Policy conflict detection
   - Policy satisfaction

3 Conclusion and future work
Contents

1 Introduction
   - Context and Goal
   - Problem
   - Our Contribution

2 Policy Configuration
   - System modeling
   - Policy modeling
   - Policy conflict detection
   - Policy satisfaction

3 Conclusion and future work
Introduction
- Context and Goal
- Problem
- Our Contribution

Policy Configuration
- System modeling
- Policy modeling
- Policy conflict detection
- Policy satisfaction

Conclusion and future work
Context and Goal

- **Security, privacy**: major issues impacting the uptake of cloud computing, particularly in public database outsourcing.

- **Industrial Database**: Data querying efficiency → Data Utility

- **Goal**: Get an optimal balance between security and functionality
Context and Goal

- **Security, privacy**: major issues impacting the uptake of cloud computing, particularly in public database outsourcing.

- Industrial Database: Data querying efficiency → Data Utility

- Goal: Get an optimal balance between security and functionality
Implementing SQL Queries over encrypted data representation.

- Fully Homomorphic Cryptosystem
  - No efficient implementation of fully homomorphic system.
- Adjustable Encryption
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].
Problem

Implementing SQL Queries over encrypted data representation.

- **Fully Homomorphic Cryptosystem**
  - No efficient implementation of fully homomorphic system.

- **Adjustable Encryption**
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].
Problem

Implementing SQL Queries over encrypted data representation.

- **Fully Homomorphic Cryptosystem**
  - No efficient implementation of fully homomorphic system.
- **Adjustable Encryption**
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].
Problem

Implementing SQL Queries over encrypted data representation.

- Fully Homomorphic Cryptosystem
  - No efficient implementation of fully homomorphic system.
- Adjustable Encryption
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

\[ a_1 \leq 100 \]
Problem

Implementing SQL Queries over encrypted data representation.

- Fully Homomorphic Cryptosystem
  - No efficient implementation of fully homomorphic system.
- Adjustable Encryption
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

\[ a_1 \leq 100 \]

→ Big Databases: The adjustment process can be quite costly.
Problem

Implementing SQL Queries over encrypted data representation.

- Fully Homomorphic Cryptosystem
  - No efficient implementation of fully homomorhopic system.
- Adjustable Encryption
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

\[ a_1 \leq 100 \]

→ Big Databases: The adjustment process can be quite costly.
→ How to prevent specific data from reaching a specific encryption state?
Problem

Implementing SQL Queries over encrypted data representation.

- Fully Homomorphic Cryptosystem
  - No efficient implementation of fully homomorphic system.
- Adjustable Encryption
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

→ Big Databases: The adjustment process can be quite costly.
→ How to prevent specific data from reaching a specific encryption state?
Implementing SQL Queries over encrypted data representation.

- **Fully Homomorphic Cryptosystem**
  - No efficient implementation of fully homomorphic system.

- **Adjustable Encryption**
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

→ Big Databases: The adjustment process can be quite costly.
→ How to prevent specific data from reaching a specific encryption state?
Implementing SQL Queries over encrypted data representation.

- **Fully Homomorphic Cryptosystem**
  - No efficient implementation of fully homomorphic system.

- **Adjustable Encryption**
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

→ Big Databases: The adjustment process can be quite costly.
→ How to prevent specific data from reaching a specific encryption state?
**Problem**

Implementing SQL Queries over encrypted data representation.

- **Fully Homomorphic Cryptosystem**
  - No efficient implementation of fully homomorphic system.
- **Adjustable Encryption**
  - The use of property-preserving encryption.
  - Adjustable “onion” encryption [Popa’11].

→ Big Databases: The adjustment process can be quite costly.
→ How to prevent specific data from reaching a specific encryption state?
→ Which encryption mechanisms to use for each attribute?
Our Contribution

- **Policy configuration**
  - Specify the policy to be applied over the outsourced data
    - Sensitive attributes
    - Security requirements
    - Utility requirements
  - Deploy the best of existing techniques allowing to get an optimal balance between security and functionality.
Current Section

1 Introduction
   ● Context and Goal
   ● Problem
   ● Our Contribution

2 Policy Configuration
   ● System modeling
   ● Policy modeling
   ● Policy conflict detection
   ● Policy satisfaction

3 Conclusion and future work
System modeling

\[ \langle \mathcal{D}, \mathcal{T}, \mathcal{A}, \mathcal{F}, \mathcal{L}, \mathcal{E} \rangle \]

- \( \mathcal{D} \): a relational database
- \( \mathcal{T} = \{ T_1, \ldots, T_n \} \): a finite set of relational tables
- \( \mathcal{A} = \{ A_{T_1}, \ldots, A_{T_n} \} \): a finite set attributes
  - \( A_{T_i} = \{ a_{1,i}, \ldots, a_{m,i} \} \) represents the set of attributes of the relational table \( T_i \)
- \( \mathcal{L} \): a finite set of security levels
- \( \mathcal{F} \): a finite set of functional requirements that can be required over the data.
- \( \mathcal{E} \): a finite set of encryption schemes
  - \( L_i \in \mathcal{L} \): a security layer that provides
  - \( F_i \subseteq \mathcal{F} \): a set of provided functional requirements
Policy modeling

Security constraints

- Confidentiality constraint
  - Defined over a relational table $T_i \in \mathcal{T}$
  - Represented by a set of attributes, e.g. $CC = \{a_1, \ldots, a_n\} \subseteq A_{T_i}$
  - States that the value assumed by the attributes in $CC$ is considered sensitive

- Security threshold constraint
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g. $TC_{a_i} = l, l \in \mathcal{L}$.
  - Four security levels can be used to classify the data
    - Top secret $\rightarrow$ RND
    - Secret $\rightarrow$ DET
    - Confidential $\rightarrow$ OPE
    - Unclassified $\rightarrow$ cleartext value
Security constraints

- **Confidentiality constraint**
  - Defined over a relational table \( T_i \in \mathcal{T} \)
  - Represented by a set of attributes, e.g.
    \[ CC = \{a_1, \cdots, a_n\} \subseteq A_{T_i} \]
  - States that the value assumed by the attributes in \( CC \) is considered sensitive

- **Security threshold constraint**
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g.
    \[ TC_{a_i} = l, l \in \mathcal{L} \]
  - Four security levels can be used to classify the data
    - Top secret \( \rightarrow \) RND
    - Secret \( \rightarrow \) DET
    - Confidential \( \rightarrow \) OPE
    - Unclassified \( \rightarrow \) cleartext value
Policy modeling

Security constraints

- Confidentiality constraint
  - Defined over a relational table \( T_i \in \mathcal{T} \)
  - Represented by a set of attributes, e.g. \( CC = \{a_1, \ldots, a_n\} \subseteq A_{T_i} \)
  - States that the value assumed by the attributes in \( CC \) is considered sensitive

- Security threshold constraint
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g. \( TC_{a_i} = l, l \in \mathcal{L} \).
  - Four security levels can be used to classify the data:
    - Top secret \( \rightarrow \) RND
    - Secret \( \rightarrow \) DET
    - Confidential \( \rightarrow \) OPE
    - Unclassified \( \rightarrow \) cleartext value
Policy modeling

Security constraints

- **Confidentiality constraint**
  - Defined over a relational table $T_i \in \mathcal{T}$
  - Represented by a set of attributes, e.g.
    $$CC = \{a_1, \cdots, a_n\} \subseteq A_{T_i}$$
  - States that the value assumed by the attributes in $CC$ is considered sensitive

- **Security threshold constraint**
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g.
    $$TC_{a_i} = l, l \in \mathcal{L}.$$  
  - Four security levels can be used to classify the data
    - Top secret $\rightarrow$ RND
    - Secret $\rightarrow$ DET
    - Confidential $\rightarrow$ OPE
    - Unclassified $\rightarrow$ cleartext value
Security constraints

- **Confidentiality constraint**
  - Defined over a relational table $T_i \in \mathcal{T}$
  - Represented by a set of attributes, e.g., $CC = \{a_1, \ldots, a_n\} \subseteq A_{T_i}$
  - States that the value assumed by the attributes in $CC$ is considered sensitive

- **Security threshold constraint**
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g., $TC_{a_i} = l, l \in \mathcal{L}$.
  - Four security levels can be used to classify the data
    - Top secret $\rightarrow$ RND
    - Secret $\rightarrow$ DET
    - Confidential $\rightarrow$ OPE
    - Unclassified $\rightarrow$ cleartext value
Policy modeling

<table>
<thead>
<tr>
<th>Security constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Confidentiality constraint</strong></td>
</tr>
<tr>
<td>Defined over a relational table $T_i \in \mathcal{T}$</td>
</tr>
<tr>
<td>Represented by a set of attributes, e.g. $CC = {a_1, \cdots, a_n} \subseteq A_{T_i}$</td>
</tr>
<tr>
<td>States that the value assumed by the attributes in $CC$ is considered sensitive</td>
</tr>
<tr>
<td><strong>Security threshold constraint</strong></td>
</tr>
<tr>
<td>Defined over an attribute</td>
</tr>
<tr>
<td>Allow the data owner to specify a security level threshold for each sensitive attribute, e.g. $TC_{a_i} = l, l \in \mathcal{L}$.</td>
</tr>
<tr>
<td>Four security levels can be used to classify the data</td>
</tr>
<tr>
<td>- Top secret $\rightarrow$ RND</td>
</tr>
<tr>
<td>- Secret $\rightarrow$ DET</td>
</tr>
<tr>
<td>- Confidential $\rightarrow$ OPE</td>
</tr>
<tr>
<td>- Unclassified $\rightarrow$ cleartext value</td>
</tr>
</tbody>
</table>
Security constraints

- **Confidentiality constraint**
  - Defined over a relational table $T_i \in \mathcal{T}$
  - Represented by a set of attributes, e.g. $CC = \{a_1, \ldots, a_n\} \subseteq A_{T_i}$
  - States that the value assumed by the attributes in $CC$ is considered sensitive

- **Security threshold constraint**
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g. $TC_{a_i} = l, l \in L$.
  - Four security levels can be used to classify the data
    - Top secret → RND
    - Secret → DET
    - Confidential → OPE
    - Unclassified → cleartext value
Policy modeling

Security constraints

- Confidentiality constraint
  - Defined over a relational table $T_i \in \mathcal{T}$
  - Represented by a set of attributes, e.g. $CC = \{a_1, \ldots, a_n\} \subseteq A_{T_i}$
  - States that the value assumed by the attributes in $CC$ is considered sensitive

- Security threshold constraint
  - Defined over an attribute
  - Allow the data owner to specify a security level threshold for each sensitive attribute, e.g. $TC_{a_i} = l, l \in \mathcal{L}$.
  - Four security levels can be used to classify the data
    - Top secret $\rightarrow$ RND
    - Secret $\rightarrow$ DET
    - Confidential $\rightarrow$ OPE
    - Unclassified $\rightarrow$ cleartext value
Utility constraint

- Defined over an attribute $a_i \in \mathcal{A}$.
- Require that some functionalities must be provided for the attribute $a_i$, e.g. $UC_{a_i} = \{f_1, \cdots, f_n\}, f_i \in \mathcal{F}$. 
Example: Banking Scenario

- $T_1(\text{SSN, Job, Address, Balance})$.
  - The account balance for a customer should always remain top secret.
  - The SSN of a customer should always remain secret.
  - All other information in the $T_1$ are unclassified.
  - Example of queries to be executed efficiently over $T_1$:
    - **Q1:**
      ```sql
      SELECT (T1_Balance + 100) 
      WHERE SSN = '321654789756'
      ```
    - **Q2:**
      ```sql
      SELECT T1_SSN 
      WHERE T1_Balance = 40000
      ```
Example: Banking Scenario

$T_1(\text{SSN, Job, Adress, Balance})$.

- The account balance for a customer should always remain top secret.
- The SSN of a customer should always remain secret.
- All other information in the $T_1$ are unclassified.

Example of queries to be executed efficiently over $T_1$:

**Q1:**
SELECT ($T_1_{\text{Balance}} + 100)$
WHERE SSN = '321654789756'

**Q2:**
SELECT $T_1_{\text{SSN}}$
WHERE $T_1_{\text{Balance}} = 40000$
Example: Banking Scenario

$T_1(\text{SSN, Job, Address, Balance})$.

- The account balance for a customer should always remain top secret.
- The SSN of a customer should always remain secret.
- All other information in the $T_1$ are unclassified.

Example of queries to be executed efficiently over $T_1$:

**Q1:**
SELECT ($T_1$\_Balance + 100)
WHERE SSN = '321654789756'

**Q2:**
SELECT $T_1$\_SSN
WHERE $T_1$\_Balance = 40000

Policy specification

$CC_{T_1} = \{\text{SSN, Balance}\}$
Example: Banking Scenario

$T_1(\text{SSN}, \text{Job}, \text{Address}, \text{Balance})$.

- The account balance for a customer should always remain top secret.
- The SSN of a customer should always remain secret.
- All other information in the $T_1$ are unclassified.

Example of queries to be executed efficiently over $T_1$:

Q1:
SELECT ($T_1$Balance + 100)
WHERE SSN = '321654789756'

Q2:
SELECT $T_1$SSN
WHERE $T_1$Balance = 40000

Policy specification

$CC_{T_1} = \{\text{SSN, Balance}\}$

$TC_{Balance} = \{\text{RND}\}$
Example: Banking Scenario

- $T_1(\text{SSN}, \text{Job}, \text{Address}, \text{Balance})$.
  - The account balance for a customer should always remain top secret.
  - The SSN of a customer should always remain secret.
  - All other information in the $T_1$ are unclassified.

- Example of queries to be executed efficiently over $T_1$:
  - Q1:
    SELECT ($T_1$\_Balance + 100)
    WHERE SSN = '321654789756'
  - Q2:
    SELECT $T_1$\_SSN
    WHERE $T_1$\_Balance = 40000

Policy specification

- $\text{CC}_{T_1} = \{ \text{SSN, Balance} \}$
- $\text{TC}_{\text{Balance}} = \{ \text{RND} \}$
- $\text{TC}_{\text{SSN}} = \{ \text{DET} \}$
Example: Banking Scenario

$T_1 (SSN, \text{Job}, \text{Address}, \text{Balance})$.

- The account balance for a customer should always remain top secret.
- The SSN of a customer should always remain secret.
- All other information in the $T_1$ are unclassified.

Example of queries to be executed efficiently over $T_1$:

Q1:
SELECT $(T_1\_\text{Balance} + 100)$
WHERE SSN = '321654789756'

Q2:
SELECT $T_1\_\text{SSN}$
WHERE $T_1\_\text{Balance} = 40000$

Policy specification

$CC_{T_1} = \{\text{SSN, Balance}\}$

$TC_{\text{Balance}} = \{\text{RND}\}$

$TC_{\text{SSN}} = \{\text{DET}\}$

$UC_{\text{Balance}} = \{\text{addition, equality}\}$
Example: Banking Scenario

- \( T_1(\text{SSN}, \text{Job}, \text{Address}, \text{Balance}) \).
  - The account balance for a customer should always remain top secret.
  - The SSN of a customer should always remain secret.
  - All other information in the \( T_1 \) are unclassified.
  - Example of queries to be executed efficiently over \( T_1 \):

Q1:
SELECT \( T_1._\text{Balance} + 100 \)
WHERE SSN = '321654789756'

Q2:
SELECT \( T_1._\text{SSN} \)
WHERE \( T_1._\text{Balance} = 40000 \)

Policy specification

\[
CC_{T_1} = \{ \text{SSN, Balance} \}
\]

\[
TC_{\text{Balance}} = \{ \text{RND} \}
\]

\[
TC_{\text{SSN}} = \{ \text{DET} \}
\]

\[
UC_{\text{Balance}} = \{ \text{addition, equality} \}
\]

\[
UC_{\text{SSN}} = \{ \text{equality} \}
\]
Policy conflict detection

- The objectives of two or more constraints cannot be simultaneously satisfied.
- Conflicts may occur between security threshold constraints and utility constraints defined over the same attribute.

**Definition**

Consider a threshold constraint $TC_a = l, l \in \mathcal{L}$ and an utility constraint $UC_a = \{f_1, \cdots, f_n\}$ defined over the attribute $a$. $TC_a$ and $UC_a$ are in conflict iff $\{f_1, \cdots, f_n\} \not\subseteq F_I$ with $F_I = \bigcup_{E_i \in \mathcal{E}} F_{E_i}$, where $l_{E_i}$ is as much secure as $l$. 
Policy conflict detection

- The objectives of two or more constraints cannot be simultaneously satisfied.
- Conflicts may occur between security threshold constraints and utility constraints defined over the same attribute.

**Definition**

Consider a threshold constraint $TC_a = l, l \in \mathcal{L}$ and an utility constraint $UC_a = \{f_1, \cdots, f_n\}$ defined over the attribute $a$. $TC_a$ and $UC_a$ are in conflict iff $\{f_1, \cdots, f_n\} \not\subseteq F_l$ with $F_l = \bigcup_{E_i \in \mathcal{E}} F_{E_i}$, where $l_{E_i}$ is as much secure as $l$. 
Example of policy conflict detection
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition, order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ addition, order search \} \]
\[ UC_{SSN} = \{ equality \} \]
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition, order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]

Encryption toolbox

AES-CBC:
- \( I_{AES} = RND \)
- \( F_{AES} = 0 \)
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition, order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]

Encryption toolbox

AES-CBC:
- \( I_{AES} = RND \)
- \( F_{AES} = 0 \)

Paillier:
- \( I_{Plir} = RND \)
- \( F_{Plir} = \{ \text{SUM, AVG, computation} \} \)
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition, order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]

Encryption toolbox

**AES-CBC:**
- \( I_{AES} = RND \)
- \( F_{AES} = \emptyset \)

**Paillier:**
- \( I_{Plr} = RND \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

**SSE:**
- \( I_{SSE} = RND \)
- \( F_{SSE} = \{ \text{like} \} \)
Example of policy conflict detection

Policy specification

$CC_{T_1} = \{SSN, Balance\}$

$TC_{Balance} = \{RND\}$

$TC_{SSN} = \{DET\}$

$UC_{Balance} = \{\text{addition, order search}\}$

$UC_{SSN} = \{\text{equality}\}$

Encryption toolbox

AES-CBC:
- $l_{AES} = \text{RND}$
- $F_{AES} = \emptyset$

Paillier:
- $l_{Plr} = \text{RND}$
- $F_{Plr} = \{\text{SUM, AVG, computation}\}$

SSE:
- $l_{SSE} = \text{RND}$
- $F_{SSE} = \{\text{like}\}$

Pohlig-Hellman:
- $l_{PH} = \text{DET}$
- $F_{PH} = \{\text{equality, join, group by}\}$
Example of policy conflict detection

Policy specification

- \( CC_{T_1} = \{ SSN, Balance \} \)
- \( TC_{Balance} = \{ RND \} \)
- \( TC_{SSN} = \{ DET \} \)
- \( UC_{Balance} = \{ addition, order \ search \} \)
- \( UC_{SSN} = \{ equality \} \)

Encryption toolbox

**AES-CBC:**
- \( I_{AES} = RND \)
- \( F_{AES} = \emptyset \)

**Paillier:**
- \( I_{Plr} = RND \)
- \( F_{Plr} = \{ \text{SUM}, \text{AVG}, \text{computation} \} \)

**SSE:**
- \( I_{SSE} = RND \)
- \( F_{SSE} = \{ \text{like} \} \)

**Pohlig-Hellman:**
- \( I_{PH} = DET \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

**Boldyreva:**
- \( I_{Bdv} = OPE \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition, order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]

First step:

Encryption toolbox

AES-CBC:
- \( l_{AES} = RND \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( l_{Plr} = RND \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

SSE:
- \( l_{SSE} = RND \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( l_{PH} = DET \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

Boldyreva:
- \( l_{Bdv} = OPE \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]

\[ TC_{Balance} = \{ RND \} \]

\[ TC_{SSN} = \{ DET \} \]

\[ UC_{Balance} = \{ addition, order search \} \]

\[ UC_{SSN} = \{ equality \} \]

First step:

\[ F_{RND} = \{ SUM, AVG, computation, like \} \]

Encryption toolbox

**AES-CBC:**
- \( l_{AES} = RND \)
- \( F_{AES} = 0 \)

**Paillier:**
- \( l_{Plr} = RND \)
- \( F_{Plr} = \{ SUM, AVG, computation \} \)

**SSE:**
- \( l_{SSE} = RND \)
- \( F_{SSE} = \{ like \} \)

**Pohlig-Hellman:**
- \( l_{PH} = DET \)
- \( F_{PH} = \{ equality, join, group by \} \)

**Boldyreva:**
- \( l_{Bdv} = OPE \)
- \( F_{Bdv} = \{ equality, join, group by, order \} \)
Example of policy conflict detection

Policy specification

\[
CC_{T_1} = \{\text{SSN, Balance}\}
\]
\[
TC_{\text{Balance}} = \{\text{RND}\}
\]
\[
TC_{\text{SSN}} = \{\text{DET}\}
\]
\[
UC_{\text{Balance}} = \{\text{addition, order search}\}
\]
\[
UC_{\text{SSN}} = \{\text{equality}\}
\]

Encryption toolbox

**AES-CBC:**
- \(l_{AES} = \text{RND}\)
- \(F_{AES} = 0\)

**Paillier:**
- \(l_{Plr} = \text{RND}\)
- \(F_{Plr} = \{\text{SUM, AVG, computation}\}\)

**SSE:**
- \(l_{SSE} = \text{RND}\)
- \(F_{SSE} = \{\text{like}\}\)

**Pohlig-Hellman:**
- \(l_{PH} = \text{DET}\)
- \(F_{PH} = \{\text{equality, join, group by}\}\)

**Boldyreva:**
- \(l_{Bdv} = \text{OPE}\)
- \(F_{Bdv} = \{\text{equality, join, group by, order}\}\)

First step:

\[
F_{\text{RND}} = \{\text{SUM, AVG, computation, like}\}
\]
\[
F_{\text{DET}} = \{\text{SUM, AVG, computation, like, equality, join, group by}\}
\]
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ \text{addition}, \text{order search} \} \]
\[ UC_{SSN} = \{ \text{equality} \} \]

Encryption toolbox

AES-CBC:
- \( l_{AES} = RND \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( l_{Plr} = RND \)
- \( F_{Plr} = \{ \text{SUM}, \text{AVG}, \text{computation} \} \)

SSE:
- \( l_{SSE} = RND \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( l_{PH} = DET \)
- \( F_{PH} = \{ \text{equality}, \text{join}, \text{group by} \} \)

Boldyreva:
- \( l_{Bdv} = OPE \)
- \( F_{Bdv} = \{ \text{equality}, \text{join}, \text{group by}, \text{order} \} \)

First step:
\[ F_{RND} = \{ \text{SUM}, \text{AVG}, \text{computation}, \text{like} \} \]
\[ F_{DET} = \{ \text{SUM}, \text{AVG}, \text{computation}, \text{like}, \text{equality}, \text{join}, \text{group by} \} \]

Second step:
\[ UC_{Balance} \subseteq F_{RND} \]
\[ UC_{SSN} \subseteq F_{DET} \]
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ addition, order search \} \]
\[ UC_{SSN} = \{ equality \} \]

First step:
\[ F_{RND} = \{ SUM, AVG, computation, like \} \]
\[ F_{DET} = \{ SUM, AVG, computation, like, equality, join, group by \} \]

Second step:
\[ UC_{Balance} \subseteq F_{RND} \quad \rightarrow \quad \text{False} \]
\[ UC_{SSN} \subseteq F_{DET} \quad \rightarrow \quad \text{True} \]

Encryption toolbox

**AES-CBC:**
- \( l_{AES} = RND \)
- \( F_{AES} = \emptyset \)

**Paillier:**
- \( l_{Plr} = RND \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

**SSE:**
- \( l_{SSE} = RND \)
- \( F_{SSE} = \{ \text{like} \} \)

**Pohlig-Hellman:**
- \( l_{PH} = DET \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

**Boldyreva:**
- \( l_{Bdv} = OPE \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Example of policy conflict detection

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ RND \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{Balance} = \{ addition, order search \} \]
\[ UC_{SSN} = \{ equality \} \]

First step:

\[ F_{RND} = \{ SUM, AVG, computation, like \} \]
\[ F_{DET} = \{ SUM, AVG, computation, like equality, join, group by \} \]

Second step:

\[ UC_{Balance} \subseteq F_{RND} \quad \rightarrow \text{False} \]
\[ UC_{SSN} \subseteq F_{DET} \quad \rightarrow \text{True} \]

\( \Rightarrow \) \( UC_{Balance} \) and \( TC_{Balance} \) are in conflict.

Encryption toolbox

AES-CBC:
- \( I_{AES} = RND \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( I_{Plr} = RND \)
- \( F_{Plr} = \{ SUM, AVG, computation \} \)

SSE:
- \( I_{SSE} = RND \)
- \( F_{SSE} = \{ like \} \)

Pohlig-Hellman:
- \( I_{PH} = DET \)
- \( F_{PH} = \{ equality, join, group by \} \)

Boldyreva:
- \( I_{Bdv} = OPE \)
- \( F_{Bdv} = \{ equality, join, group by, order \} \)
Policy satisfaction

**Goal:** Find the best combination of encryption schemes that can satisfy the set of security and utility constraints.

- A combination of encryption schemes is a subset $C \subseteq \mathcal{E}$.
- Let $C = \{E_1, \cdots, E_n\}$, $l_i$ be the security level provided by the encryption scheme $E_i$, $1 \leq i \leq n$. The security level provided by the application of $C$ is $l$ iff:
  - $l \in \{l_1, \cdots, l_n\}$
  - $\forall l_j \in \{l_1, \cdots, l_n\}, l_j$ is at least as secure as $l$
- The best combination of encryption schemes:
  - Satisfies the required utility requirements
  - Provides the highest level of protection for sensitive data
  - Contains the minimal the number of involved encryption schemes

$\Rightarrow$ NP-hard problem (minimum hyper-graph coloring problem).
Policy satisfaction

**Goal:** Find the best combination of encryption schemes that can satisfy the set of security and utility constraints.

- A combination of encryption schemes is a subset $C \subseteq E$.
- Let $C = \{E_1, \ldots, E_n\}$, $l_i$ be the security level provided by the encryption scheme $E_i$, $1 \leq i \leq n$. The security level provided by the application of $C$ is $l$ iff:
  - $l \in \{l_1, \ldots, l_n\}$
  - $\forall l_j \in \{l_1, \ldots, l_n\}, l_j$ is at least as secure as $l$
- The best combination of encryption schemes:
  - Satisfies the required utility requirements
  - Provides the highest level of protection for sensitive data
  - Contains the minimal the number of involved encryption schemes
  $\Rightarrow$ NP-hard problem (minimum hyper-graph coloring problem).
Policy satisfaction

**Goal:** Find the best combination of encryption schemes that can satisfy the set of security and utility constraints.

- A combination of encryption schemes is a subset $C \subseteq E$.
- Let $C = \{E_1, \cdots, E_n\}$, $l_i$ be the security level provided by the encryption scheme $E_i$, $1 \leq i \leq n$. The security level provided by the application of $C$ is $l$ iff:
  - $l \in \{l_1, \cdots l_n\}$
  - $\forall l_j \in \{l_1, \cdots l_n\}, l_j$ is at least as secure as $l$

- The best combination of encryption schemes:
  - Satisfies the required utility requirements
  - Provides the highest level of protection for sensitive data
  - Contains the minimal the number of involved encryption schemes

$\Rightarrow$ NP-hard problem (minimum hyper-graph coloring problem).
Policy satisfaction

**Goal:** Find the best combination of encryption schemes that can satisfy the set of security and utility constraints.

- A combination of encryption schemes is a subset $C \subseteq \mathcal{E}$.
- Let $C = \{E_1, \cdots, E_n\}$, $l_i$ be the security level provided by the encryption scheme $E_i$, $1 \leq i \leq n$. The security level provided by the application of $C$ is $l$ iff:
  - $l \in \{l_1, \cdots, l_n\}$
  - $\forall l_j \in \{l_1, \cdots, l_n\}, l_j$ is at least as secure as $l$

- The best combination of encryption schemes:
  - Satisfies the required utility requirements
  - Provides the highest level of protection for sensitive data
  - Contains the minimal number of involved encryption schemes

$\Rightarrow$ NP-hard problem (minimum hyper-graph coloring problem).
Policy satisfaction

**Goal:** Find the best combination of encryption schemes that can satisfy the set of security and utility constraints.

- A combination of encryption schemes is a subset $C \subseteq E$.
- Let $C = \{E_1, \ldots, E_n\}$, $l_i$ be the security level provided by the encryption scheme $E_i$, $1 \leq i \leq n$. The security level provided by the application of $C$ is $l$ iff:
  - $l \in \{l_1, \ldots l_n\}$
  - $\forall l_j \in \{l_1, \ldots l_n\}, l_j$ is at least as secure as $l$

**The best combination of encryption schemes:**
- Satisfies the required utility requirements
- Provides the highest level of protection for sensitive data
- Contains the minimal the number of involved encryption schemes

$\Rightarrow$ NP-hard problem (minimum hyper-graph coloring problem).
Heuristic search

- Build a solution to the problem step by step from scratch

- For each sensitive attribute $a$, choose for each iteration, the best satisfier of the chosen policy
  - Satisfies the threshold constraint defined over $a$
  - Satisfies the highest number of functionalities compared to other encryption schemes

- Complexity: polynomial time.
Policy satisfaction: Example
Policy satisfaction: Example

Policy specification

\[
CC_{T_1} = \{\text{SSN}, \text{Balance}\}
\]

\[
TC_{\text{Balance}} = \{\text{OPE}\}
\]

\[
UC_{\text{Balance}} = \{\text{computation}, \text{order},

\text{equality}, \text{AVG}, \text{group by}\}
\]

\[
TC_{\text{SSN}} = \{\text{DET}\}
\]

\[
UC_{\text{SSN}} = \{\text{equality}, \text{like}\}
\]
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{ \text{SSN, Balance} \} \]
\[ TC_{\text{Balance}} = \{ \text{OPE} \} \]
\[ UC_{\text{Balance}} = \{ \text{computation, order, equality, AVG, group by} \} \]
\[ TC_{\text{SSN}} = \{ \text{DET} \} \]
\[ UC_{\text{SSN}} = \{ \text{equality, like} \} \]

Encryption toolbox

AES-CBC:
- \( l_{AES} = \text{RND} \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( l_{Plr} = \text{RND} \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

SSE:
- \( l_{SSE} = \text{RND} \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( l_{PH} = \text{DET} \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

Boldyreva:
- \( l_{Bdv} = \text{OPE} \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ OPE \} \]
\[ UC_{Balance} = \{ computation, order, equality, AVG, group by \} \]
\[ TC_{SSN} = \{ DET \} \]
\[ UC_{SSN} = \{ equality, like \} \]

Attribute Balance:

Encryption toolbox

AES-CBC:
- \( I_{AES} = RND \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( I_{Plr} = RND \)
- \( F_{Plr} = \{ SUM, AVG, computation \} \)

SSE:
- \( I_{SSE} = RND \)
- \( F_{SSE} = \{ like \} \)

Pohlig-Hellman:
- \( I_{PH} = DET \)
- \( F_{PH} = \{ equality, join, group by \} \)

Boldyreva:
- \( I_{Bdv} = OPE \)
- \( F_{Bdv} = \{ equality, join, group by, order \} \)
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{ \text{SSN, Balance} \} \]
\[ TC_{Balance} = \{ \text{OPE} \} \]
\[ UC_{Balance} = \{ \text{computation, order, equality, AVG, group by} \} \]
\[ TC_{SSN} = \{ \text{DET} \} \]
\[ UC_{SSN} = \{ \text{equality, like} \} \]

Attribute Balance:

Encryption toolbox

AES-CBC:
- \( l_{AES} = \text{RND} \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( l_{Plr} = \text{RND} \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

SSE:
- \( l_{SSE} = \text{RND} \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( l_{PH} = \text{DET} \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

Boldyreva:
- \( l_{Bdv} = \text{OPE} \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Policy satisfaction: Example

Policy specification

$CC_{T_1} = \{ \text{SSN, Balance} \}$

$TC_{\text{Balance}} = \{ \text{OPE} \}$

$UC_{\text{Balance}} = \{ \text{computation, order, equality, AVG, group by} \}$

$TC_{\text{SSN}} = \{ \text{DET} \}$

$UC_{\text{SSN}} = \{ \text{equality, like} \}$

Attribute $\text{Balance}$:
- Boldyreva

Encryption toolbox

AES-CBC:
- $I_{AES} = \text{RND}$
- $F_{AES} = \emptyset$

Paillier:
- $I_{Plr} = \text{RND}$
- $F_{Plr} = \{ \text{SUM, AVG, computation} \}$

SSE:
- $I_{SSE} = \text{RND}$
- $F_{SSE} = \{ \text{like} \}$

Pohlig-Hellman:
- $I_{PH} = \text{DET}$
- $F_{PH} = \{ \text{equality, join, group by} \}$

Boldyreva:
- $I_{Bdv} = \text{OPE}$
- $F_{Bdv} = \{ \text{equality, join, group by, order} \}$
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{ \text{SSN, Balance} \} \]
\[ TC_{Balance} = \{ \text{OPE} \} \]
\[ UC_{Balance} = \{ \text{computation, order, equality, AVG, group by} \} \]
\[ TC_{SSN} = \{ \text{DET} \} \]
\[ UC_{SSN} = \{ \text{equality, like} \} \]

Attribute Balance:
- Boldyreva

Encryption toolbox

AES-CBC:
- \( l_{AES} = \text{RND} \)
- \( F_{AES} = \emptyset \)

Paiillier:
- \( l_{Plr} = \text{RND} \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

SSE:
- \( l_{SSE} = \text{RND} \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( l_{PH} = \text{DET} \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

Boldyreva:
- \( l_{Bdv} = \text{OPE} \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Policy specification

\[ CC_{T_1} = \{SSN, Balance\} \]
\[ TC_{Balance} = \{OPE\} \]
\[ UC_{Balance} = \{computation, order, equality, AVG, group by\} \]
\[ TC_{SSN} = \{DET\} \]
\[ UC_{SSN} = \{equality, like\} \]

Attribute Balance:
- Boldyreva
- Paillier

Encryption toolbox

AES-CBC:
- \( I_{AES} = RND \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( I_{Plr} = RND \)
- \( F_{Plr} = \{SUM, AVG, computation\} \)

SSE:
- \( I_{SSE} = RND \)
- \( F_{SSE} = \{like\} \)

Pohlig-Hellman:
- \( I_{PH} = DET \)
- \( F_{PH} = \{equality, join, group by\} \)

Boldyreva:
- \( I_{Bdv} = OPE \)
- \( F_{Bdv} = \{equality, join, group by, order\} \)
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{SSN, Balance\} \]
\[ TC_{Balance} = \{OPE\} \]
\[ UC_{Balance} = \{\text{computation, order, equality, AVG, group by}\} \]
\[ TC_{SSN} = \{DET\} \]
\[ UC_{SSN} = \{\text{equality, like}\} \]

Attribute Balance:
- Boldyreva
- Paillier

Attribute SSN:

Encryption toolbox

AES-CBC:
- \( I_{AES} = \text{RND} \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( I_{Plr} = \text{RND} \)
- \( F_{Plr} = \{\text{SUM, AVG, computation}\} \)

SSE:
- \( I_{SSE} = \text{RND} \)
- \( F_{SSE} = \{\text{like}\} \)

Pohlig-Hellman:
- \( I_{PH} = \text{DET} \)
- \( F_{PH} = \{\text{equality, join, group by}\} \)

Boldyreva:
- \( I_{Bdv} = \text{OPE} \)
- \( F_{Bdv} = \{\text{equality, join, group by, order}\} \)
Policy satisfaction: Example

Policy specification

\[ CC_{T_1} = \{ SSN, Balance \} \]
\[ TC_{Balance} = \{ OPE \} \]
\[ UC_{Balance} = \{ \text{computation, order, equality, AVG, group by} \} \]
\[ TC_{SSN} = \{ \text{DET} \} \]
\[ UC_{SSN} = \{ \text{equality, like} \} \]

Attribute Balance:
- Boldyreva
- Paillier

Attribute SSN:
- Pohlig-Hellman
- SSE

Encryption toolbox

AES-CBC:
- \( I_{AES} = \text{RND} \)
- \( F_{AES} = \emptyset \)

Paillier:
- \( I_{Plr} = \text{RND} \)
- \( F_{Plr} = \{ \text{SUM, AVG, computation} \} \)

SSE:
- \( I_{SSE} = \text{RND} \)
- \( F_{SSE} = \{ \text{like} \} \)

Pohlig-Hellman:
- \( I_{PH} = \text{DET} \)
- \( F_{PH} = \{ \text{equality, join, group by} \} \)

Boldyreva:
- \( I_{Bdv} = \text{OPE} \)
- \( F_{Bdv} = \{ \text{equality, join, group by, order} \} \)
Current Section

1. Introduction
   - Context and Goal
   - Problem
   - Our Contribution

2. Policy Configuration
   - System modeling
   - Policy modeling
   - Policy conflict detection
   - Policy satisfaction

3. Conclusion and future work
Conclusion and future work

- We present a set of algorithms allowing to:
  - Analyze security and utility requirements to detect possible conflicts
  - decide about the best acceptable trade-off between functionality and security requirements

- Future work
  - Combine encryption base mechanisms with other kind of security mechanisms, such as anonymization, watermarking, etc...
Questions