Secure Data Exchange and Data Leakage Detection in Untrusted Cloud

Denis Ulybyshev, Bharat Bhargava, Aala Alsalem
Computer Science Department, CERIAS
Purdue University, West Lafayette, USA
Outline

- Problem Statement
- Related Work
- Core Design
- Evaluation
- Contributions
Problem Statement

Secure Data Exchange / Leakage Detection

• Authorized service can only access data items for which it is authorized
• Data exchange model must consider context and client's attributes
• Detect data leakages made by insiders to unauthorized services
• Measure data leakage (what got leaked, when, to where, how sensitive was the data)
## Recent Data Leakages Examples

<table>
<thead>
<tr>
<th>Company</th>
<th>Time</th>
<th>Incident Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthem</td>
<td>Feb.2015</td>
<td>78.8 million of PII records got leaked</td>
</tr>
<tr>
<td>Experian Information Solutions and T-Mobile, USA</td>
<td>Sep.2015</td>
<td>Data (SSN, credit card information) of about 15 million customers who applied for credit got leaked</td>
</tr>
</tbody>
</table>
Problem Statement

Leakage of Medical Info

Scenario of EHR Dissemination in Cloud (proposed by Dr. Leon Li, NGC)
Active Bundle (AB) parts [17], [18]

- **Sensitive data:**
  - Encrypted data items

- **Metadata:** describe AB and its access control policies
  - Policies [21], [22] manage AB interaction with services and hosts

- **Policy Engine** [26]: enforces policies specified in AB
  - Provides tamper-resistance of AB [1]
Tamper Resistance of AB

- Key is not stored inside AB [2]
- Separate symmetric key is used for each separate data value
- Ensure protection against tampering attacks

Aggregation\{d_i\}
\(\text{Execution info; Digest(AB Modules); Resources}\)

Aggregation\{d_i\} (Tampered (\text{Execution info; Digest(AB Modules); Resources}))

Denis Ulybyshev
Framework Architecture

1. Client requests data from EHR DB
2. Service (authenticated client) requests trust and leakage verification from CM
3. CM responses with trust level of requesting service and leakage check result
4. Data request is transferred to EHR, Access control policies are evaluated
5. Result is sent to client

Client_1
Web Crypto Authentication

Client_2
Web Crypto Authentication

Client_M
Web Crypto Authentication
Data dissemination features

Data Dissemination based on [3]:

- Access control policies [27]
- Trust level of a subject (service, user)
- Context (e.g. emergency vs. normal)
- Security level of client’s browser (crypto capabilities) [23], [24]
- Authentication method (password-based, fingerprint etc)
- Source network (secure intranet vs. unknown network)
- Type of client’s device: desktop vs. mobile (detected by Authentication Server)
Attribute and role–based data dissemination

AUTHENTICATED CLIENT

Browser’s Crypto Level: High
Authentication Method: Fingerprint
Client’s device: Desktop
Source network: Corporate Intranet
Role: Doctor

AUTHENTICATED CLIENT

Browser’s Crypto Level: Low
Authentication Method: Password
Client’s device: Mobile
Source network: Unknown
Role: Insurance Agent

Email

ACCESSIBLE TEXT

Email

INACCESSIBLE TEXT 1
ACCESSIBLE TEXT
INACCESSIBLE TEXT 2
How can data get leaked by authorized subject?

- In the form of encrypted data (the whole AB is leaked):
  - Data is protected by AB, but fact of leakage can be detected
  - Detection Phase 1: digital watermark [12] can be checked by web crawler to detect copyright violations
  - Detection Phase 2: based on Obligations: how data is used by authorized party?
    - *Obligations are enforced by Central Monitor (TTP)*
    - *CM checks whether data is supposed to be where they are*
Core Design: Data Leakage Detection

Service X is authorized to read $d_1$ from AB

- Enc [Data(D)] = \{Enc_{k_1}(d_1), \ldots , Enc_{k_n}(d_n)\} 
- Access Control Policies (P) = \{p_1, \ldots , p_k\}

- Service X is authorized to read $d_1$ from AB
- Service X may leak decrypted $d_1$ or the entire AB to Y

AB contains:

Cent ral Monitor

$P$

Src ID (of X)
Dest ID (of Y)
Time
Class (Type) of data $d_1$
Core Design: Data Leakage Detection

• When service tries to decrypt AB data, CM is notified about that: “Service Y tries to decrypt $d_1$ arrived from X“
• If CM is unreachable, decryption terminates
• CM checks against centralized Obligations DB: whether $d_1$ is supposed to be at Y. If NO then:
  • Blacklist X, Y
  • Reduce their trust level
  • Mark data $d_1$ as compromised and notify services about it
  • Raise the level of $d_1$ classification
How can data get leaked by authorized subject?

- In the form of decrypted (raw) data:
  - Data is not protected by AB anymore
  - Detection based on visual / digital watermarks embedded into data
Plaintext Data Leakage Mitigation Methods

- **Layered Approach:** Don't give all the data to the requester at once
  - First give part of data (incomplete, less sensitive)
  - Watch how it is used and monitor trust level of using service
  - If trust level is sufficient – give next portion of data
- **Raise the level of data classification** to prevent leakage repetition
- **Intentional leakage** to create uncertainty and lower data value
- **Use provenance data stored at CM** to identify the list of suspects
- **Monitor network messages**
  - Check whether they contain e.g. credit card number that satisfies specific pattern and can be validated using regular expressions [25]
Data Leakage Damage Assessment

• After data leakage is detected damage is assessed based on:
  • To whom was the data leaked (unknown service with low trust level vs. service with high level of trust)
  • Sensitivity (Classification) of leaked data (classified vs. unclassified)
  • When was leaked data received (recent or old data)
  • Can other sensitive data be derived from the leaked data (i.e. diagnosis can be derived from leaked medical prescriptions)

\[
\text{Damage} = K(\text{Data is Sensitive}) \times K(\text{Service is Malicious}) \times F(t)
\]

, where \(F(t)\) is the data sensitivity function in time
Timing of Leaked Data

- Data-related event (e.g. final exam) occurs at $t_0$
- Threat from data being leaked before $t_0$ is high
- Threat from data being leaked after $t_0$:
  1) No threat at all
  2) Linearly decreases with time
  3) Remains constant (for highly-sensitive data)
Evaluation

Performance overhead for EHR, hosted locally

Performance overhead for EHR, hosted by Google Cloud
Performance overhead imposed by data leakage detection capabilities
Contributions

Contributes to Data Confidentiality and Integrity

- Dissemination does not require data owner’s availability
- Trust level of subjects is constantly recalculated
- On-the-fly key generation
- Supports data updates for multiple subjects
- Supports attribute-based data dissemination. Attributes include cryptographic capabilities of client’s browser [28]
- Tamper-resistance: data and policies integrity is provided
- Data leakage detection and leakage damage assessment
- Captures data provenance for use in leakage measure and forensics
References

10. SandMark: A Tool for the Study of Software Protection Algorithms
http://cgi.cs.arizona.edu/~sandmark/sandmark.html
DCT", 3rd Intl. Conf. on Environmental Science and Information Application Technology,
13. M. Stamp, "Digital Rights Management: The Technology Behind The Hype," Journal of
accessed:Mar.2018
17. L. Ben Othmane and L. Lilien, “Protecting privacy in sensitive data dissemination with
active bundles,” 7-th Annual Conf. on Privacy, Security and Trust (PST 2009), Saint John, New
Brunswick, Canada, Aug. 2009, pp. 202-213
18. L. Lilien and B. Bhargava, ”A scheme for privacy-preserving data dissemination,” IEEE
2006, pp. 503-506.
19. B. Bhargava, “Secure/resilient systems and data dissemination/provenance,” NGCRC Project Proposal, CERIAS, Purdue University, Mar. 2018
https://www.dropbox.com/s/30scw1srqsmyq6d/BhargavaTeam_DemoVideo_Spring16.wmv?dl=0
22. “eXtensible access control markup language (XACML) version 3.0,”
http://www.w3.org/TR/webauthn, accessed: Mar. 2018
25. “Finding or verifying credit card numbers,”