Minimizing Trust Liabilities in Secure Messaging Infrastructures

Himanshu Khurana
NCSA, University of Illinois

CERIAS Seminar, Purdue University
August 31, 2005
Introduction

- **Collaborative applications need a messaging infrastructure**
  - E.g., Groupware apps use email, conferencing uses group communication, tickers (stock, news) uses pub/sub

- **Widespread use requires secure messaging infrastructures**
  - *Integrity* and *authentication* typically provided via CA/PKI
    - Works but imposes certificate distribution/revocation problems
  - *Confidentiality* provided by trusted servers
    - Servers bear significant trust liability of maintaining confidentiality of messages and keys
      - E.g., group controllers store long term and session keys
  - *Availability* provided via replication
    - However, replicating keys makes the system insecure
Introduction

- **Importance of minimizing trust liability**
  - Servers attacked and compromised; e.g., Summer’04
  - Electronic crime on the rise

- **Challenges for minimizing trust liability**
  - Infrastructure servers must not be able to access messages
    - However, servers often need to process these messages
  - Solution must scale to support a large number of users
    - No establishment of keys between entities - $O(n^2)$ problem

- **Approach**
  - Explore novel encryption techniques to address the problem
  - Use techniques to design secure messaging infrastructures
    - Group communication, Publish/Subscribe, Email messaging infrastructures
Introduction to Proxy Encryption

- **Basic idea**
  - Convert ciphertext for one key into ciphertext for another key without revealing secrets keys or cleartext messages

- **Example construction based on El Gamal**
  - Keys for A, B: (SK_A, PK_A), (SK_B, PK_B)
  - Proxy key for transformation agent T: $\pi = (SK_B - SK_A)$
  - Encryption of m for Alice: $g^r, m(PK_A)^r$
  - Transformation of message for Bob by T: $g^r, m.(PK_A)^r.(g^r)^\pi = g^r, m.g^{r(SK_A+SK_B- SK_A)} = g^r, m(PK_B)^r$

- **Applications**
  - Key Escrow without revealing secret keys
  - Smartcard key management, file sharing
Introduction to E-mail List Services

- **ELSs** enable users to easily exchange emails
  - LS bears all the overhead

- Increasingly popular for exchange of both public and private content ⇒ security is an important concern
  - E.g., there are over 300,000 registered lists on LISTSERV while only 20% of them serve public content

- Little or no work in providing security solutions for ELSs
  - We provide SELS: Secure Email List Service
    - solutions for confidentiality, integrity, and authentication

**List Server (LS)**
- creates lists
- forwards emails
- archives email

**List Moderator (LM)**
- creates lists
- Subscribes users

**User/subscriber**
- subscribes to lists
- sends/receives email
Security Solutions

- **Confidentiality**: only authorized users (i.e. list subscribers) should be able to read emails – *list server is excluded*

- **Integrity**: receivers must be assured that email has not been modified

- **Authentication**: receivers must be able to verify the sender’s identity
Need for SELS

- **Maintain privacy of sensitive information**
  - Security lists concerning infrastructure protection
    - Subscribers: system/security administrators
  - Executives lists concerning corporate strategies
    - Subscribers: managers, executives

- **Enforce privacy laws regarding user data**
  - Multi-institution R&D on healthcare needs to enforce HIPAA laws

- **Enable trustworthy groupware applications**
  - Document annotation and storage
  - Distributed software development
  - Mobile Teamwork
Pretty Good Privacy (PGP) Security for Two-party Email Exchange

Alice ——— Base-64 encoded message ——— Bob

<table>
<thead>
<tr>
<th>Header in Plaintext</th>
<th>Encrypt_k(m, Sig(m))</th>
<th>Encrypt (k) w/ PK_B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plaintext m</td>
<td>Sign h(m) with SK_A</td>
<td>RSA, El Gamal</td>
</tr>
<tr>
<td></td>
<td>(RSA, DSA)</td>
<td></td>
</tr>
</tbody>
</table>

PGP: Bob’s encryption key PK_B

PGP: Alice’s signature verification key PK_A

- Other approaches for achieving confidentiality, integrity, and authentication
  - Privacy Enhanced Mail (PEM), Secure MIME (S/MIME)
    - Challenge: certificate distribution and validation
  - Identity-based encryption
    - Public key (for encryption) generated from email address
Satisfying Security in ELSs

- **Confidentiality**

  - Users exchange symmetric key out-of-band for encryption
    - Difficult to provide secure key distribution
  - Users use PGP web-of-trust to exchange secure email
    - High key management overhead for each user
  - Use cryptographic hardware to do decryption/encryption
    - Expensive, broken in the past (e.g., CCA on IBM 4758)

- **Possible Solutions**

  - Problem: confidentiality requirement not met; adversary can compromise LS to obtain e-mails

- **Our solution**

  - Use software-based proxy re-encryption at LS to transform encrypted e-mail between sender and receivers without requiring access to plaintext
Establish key pair $(K_{U1}, PK_{U1})$

LM, LS implicitly agree

$K_{LK} = K_{LM} + K_{LS}$ is list key

Assumption

- LM is an independent entity not controlled by LS
Sending Secure E-mails in SELS

**Key Store:** (SKₐ, PKₐ)

**Key Store:** Members’ corresponding private keys K'ₜₐ

**Alice → LS**

<table>
<thead>
<tr>
<th>Email Header</th>
<th>Encrypt (m, Sig(m)) w/ k</th>
<th>Encrypted k w/ PKₐ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Plaintext m</td>
<td>Sig(m) w/ SKₐ (RSA, DSA)</td>
<td>(AES, 3DES) (SELS/El Gamal)</td>
</tr>
</tbody>
</table>

**Bob ← LS**

<table>
<thead>
<tr>
<th>Email Header</th>
<th>Encrypt (m, Sig(m)) w/ k</th>
<th>Transform k W/ K'ᵤₐ, K'ᵤₜ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email Plaintext m</td>
<td>Sig(m) w/ SKₐ (RSA, DSA)</td>
<td>(AES, 3DES) (SELS Proxy Re-encryption)</td>
</tr>
</tbody>
</table>

k.g^{K_Br} mod p = k.g^{K_Ar}.g^{K'ₜₐr}.(g^{K'Br})^{-1} mod p
Recent and Ongoing Work: Formal Verification and Implementation

- **Formal Verification with Proverif**
  - Fully automated protocol verification tool
  - SELS provides confidentiality even if LS is compromised

- **Implementing SELS Prototype**
  - *Client-side:* Integrate web-based subscribe functions with GnuPG to generate standard PGP keys
    - Any email client that employs a PGP plugin can use our tools
    - *No modifications to any email client needed*
  - *Server-side:* Develop plugins for Majordomo and Mailman
  - Development tools
    - GnuPG toolkit, libgcrypt library, bouncycastle, C/C++, Java
Introduction to Pub/Sub

Pub/Sub Infrastructure (e.g., Gryphon, Siena)

- **Border Broker**
- **Broker**
- **Publisher**
- **Subscriber**

- **Applications**: software updates, location-based services for wireless networks, supply chain management, traffic control, and stock quote dissemination
- **Three types**: Topic-based, type-based, and content-based
  - Content-based considered to be the most general
Security Challenges Addressed for Content-Based Pub/Sub Systems (CBPS)

- **Confidentiality**
  - Deliver information to authorized subscribers
  - PBs and SBs don’t trust broker network

- **Usage-based accounting**
  - E.g., for stock quote dissemination

- **Solution Highlights**
  - Strong adversarial model: PB/SBs don’t trust broker network
  - No security associations needed between PBs and SBs
  - No modifications to existing matching & routing algorithms
Confidentiality

- Adversary has access to network traffic ⇒ contents cannot be disclosed to brokers

- One approach: perform computations on encrypted data
  - Difficult to implement in practice
  - Require modifications to matching and routing techniques

- Observation
  - Only selected parts of an event’s content need to be confidential
  - Matching and routing can be accomplished without these parts

- Our Approach
  - Encode events in XML documents
  - Selectively encrypt sensitive parts of events
    - Use Bertino and Ferrari’s XML document dissemination techniques
  - Distribute keys to authorized subscribers
    - Use Jakobsson’s proxy encryption techniques
Confidentiality Examples

Cleartext Event Contents

Encrypt

Encrypted Packages

Message: id 100
<?xml?><stock>
  <symbol>YHOO</symbol>
  <price>70.2</price>
  <open>50</open>
  <volume>10000</volume>
</stock>

Message: id 100
<?xml?><stock>
  <symbol>YHOO</symbol>
  <price>E_k(70.2)</price>
  <open>50</open>
  <volume>10000</volume>
</stock>

Message: id 200
<?xml?><gamescore>
  <date>8/5/04</date>
  <teams>NY-CA</teams>
  <score>10-3</score>
</gamescore>

Message: id 200
<?xml?><gamescore>
  <date>8/5/04</date>
  <teams>NY-CA</teams>
  <score>E_k(10-3)</score>
</gamescore>

E_k() → symmetric key encryption (e.g., AES) using key k
Enc_{PK}() → El Gamal public key encryption using key PK
Distributing Keys to Authorized Subscribers

Proxy Security and Accounting Service (PSAS)

\[ \text{RSA Signature Key } (K_{PS}, PK_{PS}): K_{ps} = \sum_{i=1}^{m} K_{PS_i} \text{ where } K_{PS_i} \text{ is a key share held by any coordinator} \]

\[ \text{For each EG decryption key } (K_{PS}, PK_{PS}): K_{ps} = \sum_{i=1}^{t} K_{PS_i} \text{ where } K_{PS_i} \text{ is a key share held by any server} \]
Protocol Overview

Event publication, routing, and delivery

- Coordinators verify request
- Generate and sign k tx task
- Servers decrypt k with $K_{PS}$
- Servers encrypt k with $PK_{SB}$

- Transformation process produces a verifiable certificate
- Used to provide usage-based accounting
Scalability

Per subscriber event delivery
Network Delay 0.35 msec

$S$, Maximum Number of Subscribers

Time for an exponentiation $T_{exp}$ (msec)
Secure Group Communication (SGC)

- SGC needed to support many military and commercial applications; e.g.,
  - Conferencing (Video and/or Audio), Command-and-Control Systems, Interactive Distance-Learning

- Group Key Management (GKM) cornerstone of SGC
  - Involves distribution of symmetric key to group members
  - Must be efficient and scalable
    - Shared key changed every time a member joins/leaves group
  - Existing GKM Schemes
    - Logical Key Hierarchies (LKH) using Group Controllers (GC)
      - Advantage: Very efficient, constant number of rounds
      - Drawback: GC is completely trusted
    - Decentralized or Contributory Schemes
      - Advantage: Does not involve a GC
      - Drawback: Do not scale as well, impose higher overhead
TASK - Tree-based w/ Asymmetric Split Keys

- **Efficient and Scalable**
  - log(n) computation and storage
  - log(n) message sizes and constant number of rounds

- **Partially Trusted GC**
  - GC does not store encryption keys
    - Confidentiality maintained even if GC is compromised
    - Therefore, GC no longer single point of security failure
  - Instead, GC uses proxy encryption to transform messages between members for key establishment
  - Simpler recovery from GC compromise

- **Assumptions**
  - All protocol messages are signed
  - GC and a member are not simultaneously compromised
**Difference between LKH & TASK**

![Diagram showing the structure of LKH and TASK with keys and members labeled.]

**LEGEND**
- $k_i$: symmetric key held by member(s) and GC
- Session key (Data Encrypting Key - DEK) = $k_{1,9}$
- $K_{i}$: asymmetric private key held by member(s)
- $K'_{i}$: corresponding asymmetric private key held by GC
- $(K_i + K'_{i}) \mod q = GKEK$ - the Group Key Encrypting Key
- Session key (Data Encrypting Key - DEK) = $h(K_{1,9})$
Member Leave Details

Proxy encryption:
\(\{r\}_{gK_{M7}} = g^s, r.(g^{K_{M8}})^s.g^{sK_{M8}}.(g^{sK_{M8}})^{-1}\)
Member Join Protocol

- Sponsor uses current session key to encrypt $r$
  - Members decrypt $r$, update private keys
  - Sponsor sends keys to joining user
  - Joining member contributes share to its private key
## LKH versus TASK Costs

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Entity</th>
<th>Events</th>
<th>Rounds</th>
<th>Communication</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK</td>
<td>GC</td>
<td>Join</td>
<td>2</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>2</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Member</td>
<td></td>
<td>Join</td>
<td>2</td>
<td>O(h)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>2</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td>LKH</td>
<td>GC</td>
<td>Join</td>
<td>1</td>
<td>O(h)</td>
<td>0</td>
</tr>
<tr>
<td>(Wong et al)</td>
<td></td>
<td>Leave</td>
<td>1</td>
<td>O(dh)</td>
<td>0</td>
</tr>
<tr>
<td>Member</td>
<td></td>
<td>Join</td>
<td>1</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>1</td>
<td>N/A</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Entity</th>
<th>Events</th>
<th>Rounds</th>
<th>Communication</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>TASK</td>
<td>GC</td>
<td>Join</td>
<td>2</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>2</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Member</td>
<td></td>
<td>Join</td>
<td>2</td>
<td>O(h)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>2</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td>LKH</td>
<td>GC</td>
<td>Join</td>
<td>1</td>
<td>O(h)</td>
<td>0</td>
</tr>
<tr>
<td>(Wong et al)</td>
<td></td>
<td>Leave</td>
<td>1</td>
<td>O(dh)</td>
<td>0</td>
</tr>
<tr>
<td>Member</td>
<td></td>
<td>Join</td>
<td>1</td>
<td>O(1)</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leave</td>
<td>1</td>
<td>N/A</td>
<td>0</td>
</tr>
</tbody>
</table>

**LEGEND**
- n: number of current group members
- d: degree of tree
- h: height of tree
Challenges for Proxy Encryption

- **Proxy key generation without trusted third party**
  - Ateniese et al. use a novel IBE-based scheme
    - Requires (1) resource owner, (2) users to manage two private keys
  - We use a distributed key generation process
    - Requires additional communication round

- **Resilience to collusion**
  - Simultaneous compromise of member and GC reveals GKEK
    - Exploring solutions
Conclusions

- Confidentiality is needed in secure messaging inf.
  - Solutions today use trusted servers
  - Servers bear significant trust liabilities
- Proxy encryption techniques provide novel solutions

<table>
<thead>
<tr>
<th>Infrastructure</th>
<th>Latencies</th>
<th>Group Membership</th>
<th>Proxy Enc</th>
<th>Solution Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>secs/ mins</td>
<td>Known</td>
<td>Per msg</td>
<td>Low</td>
</tr>
<tr>
<td>GCS</td>
<td>msecs</td>
<td>Known</td>
<td>Key Dist</td>
<td>Medium?</td>
</tr>
<tr>
<td>Pub/Sub</td>
<td>msecs/ secs</td>
<td>Dynamic</td>
<td>Per msg</td>
<td>High</td>
</tr>
</tbody>
</table>
Questions?

- Papers available at http://ncsa.uiuc.edu/people/hkhurana

- Contact
  - hkhurana@ncsa.uiuc.edu