CSC/ECE 774 Advanced Network Security

Topic 3.3: Fair Exchange

Outline

- Overview of Fair Exchange
- Optimistic Fair Exchange
  - A General Protocol
  - Optimized Protocol
    - Contract signing
- Self-study
  - Optimized Protocols
    - Certified mail
    - Payment for receipt
    - Fair purchase

Fair Exchange

- A fair exchange should guarantee that at the end of the exchange
  - Either each party has received what it expects to receive,
  - Or no party has received anything
- Examples
  - Certified mail
  - Contract signing
  - Payment
Traditional Fair Exchange

- ISO proposals
  - Use a TTP to ensure fairness
- Limitations
  - TTP is heavily involved
  - Bottleneck
  - Single point of failure

Optimistic Fair Exchange

- Assumptions
  - Most participants are honest
- Allow participants to exchange without TTP
- Fall back to TTP when there are failures
  - Dishonest participants, communication failures, etc.

Three Phases of Optimistic Fair Exchange

- Phase 1
  - The parties try to exchange items without a TTP
- Phase 2
  - The parties try to exchange items through a TTP
- Phase 3
  - Each computer outputs all evidence and any participant may visit a court
Degree of Fairness

- **Strong (true) fairness**
  - If the TTP is able to
    - Undo a transfer of an item (revocability)
      - Example: revoke a signed contract
    - Produce a replacement for it (Generatability)
      - Example: generate a replacement of a receipt

- **Weak fairness**
  - If the TTP can only produce affidavits
  - Requires an external dispute resolution system
    - Example: court

Generic Exchange Protocol

- **Two stages**
  - Stage 1 (Two flows)
    - The originator O and the recipient R promise each other an exchange of items
  - Stage 2 (Three flows)
    - Exchange the items along with non-repudiation tokens

Notations

- item\(_x\): the item X wants to send
- descr\(_x\): a description of item\(_x\)
- expect\(_x\)(descr\(_x\), descr\(_y\)):
  - Evaluate to true if X is satisfied with exchanging item\(_x\) with item\(_y\).
- fits(descr, item)
  - Evaluate to true if the description fits the item
- h(): hash function
- \((key, comm) = commit(item)\)
  - Generate a commitment comm to item, and also generate a key, without which it’s impossible to get the item.
- Verifiable encryption.
- \(open(item, key, comm)\)
  - Use key to open the item whose commitment is comm.
Generic Exchange Protocol (Cont’)

In: item, descr, expect

Choose y_R (recovery authenticator) randomly; determine T
(key_T, com_T) := commit (item)

m_R := sign_R (T, R, h(y_R), h(r_R), T, com_T, descr_R)

If not expect_R (descr_R, descr_O) then
Abort;
Choose y_O (recovery authenticator) randomly;
(key_O, com_O) := commit (item)

m_O := sign_O (O, h(m_1), h(y_O), h(r_O), com_O, descr_O)

m_1 := sign_R (O, h(m_1), h(y_R), h(r_R), com_R, descr_R)

m_2 := sign_O (O, h(m_2), h(y_O), h(r_O), com_O, descr_O)

Output:
item_R NRO Token: (m_2, key_R, com_R)
NRR Token: (m_1, m_2, r_R)

Output:
item_O NRO Token: (m_1, key_O, com_O)
NRR Token: (m_1, m_2, r_O)

• Question:
  – Why can these tokens guarantee NRO or NRR?
Recovery for O

\[
\begin{align*}
\text{O} & \quad \text{T} & \quad \text{R} \\
\text{m} \geq m_1, m_2, y & \quad & \text{m}_T = \text{sign}_T(h(m)) \\
\end{align*}
\]

If [the received messages fit together] then

retransmit \( m_\mathcal{O} \), observable by \( T \)

else if [timeout] then

\[ m_\mathcal{O} \]

\[ m_\mathcal{T} = \text{sign}_T(h(m)) \]

else

open (item \( \mathcal{O} \), key \( \mathcal{O} \), com \( \mathcal{O} \))?

fits (item \( \mathcal{O} \), descr \( \mathcal{O} \))?

\[ m_\mathcal{O} \]

\[ m_\mathcal{T} = \text{sign}_T(h(m)) \]

Question

• Can this recovery protocol guarantee
  – Strong fairness for \( \mathcal{O} \)?
  – Weak fairness for \( \mathcal{O} \)?
  
•

Recovery for R

\[
\begin{align*}
\text{O} & \quad \text{T} & \quad \text{R} \\
\text{m} \geq m_1, m_2, y & \quad & \text{m}_T = \text{sign}_T(h(m)) \\
\end{align*}
\]
Question

- Can this recovery protocol guarantee
  - Strong fairness for R?
  - Weak fairness for R?

Types of Items

- Confidential data
  - Data that will be released during the protocol
  - Example: Software

- Public data
  - Data that will be released even if the protocol execution fails
  - Purpose: fair exchange of non-repudiation tokens.
  - Example: contract

- Payments
  - A payment sub-protocol that is executed to transfer value from payer to payee
  - Example: PayWords

Types of Items (Cont’d)

- Generatable
  - The TTP can produce a replacement of the item

- Revocable
  - The TTP can undo the transfer of the item
Exchange Types

<table>
<thead>
<tr>
<th>Public Data</th>
<th>Conf. Data</th>
<th>Payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contract</td>
<td>Certified</td>
<td>Payment</td>
</tr>
<tr>
<td>Signing</td>
<td>Mail</td>
<td></td>
</tr>
</tbody>
</table>

Optimized Protocol -- Contract Signing

**O**

\(\text{In: contract}_1\)

**T**

\(\text{In: contract}_2\)

Choose \(o_R\) randomly; determine \(T\)

\(m_1 \leftarrow \text{sign}_O(T, R, h(o_R), t, \text{contract}_1)\)

Choose \(y_R\) randomly; contract\(_3\) = contract\(_1\)

\(m_2 \leftarrow \text{sign}_R(h(m_1), h(y_R))\)

\(m_3 \leftarrow o_O\)

Contract Signing (Cont’d)

\[\text{If [timeout] then} \]

\[m_t \leftarrow m_t, m_2, y_R\]

\[\text{If [the received messages fit together] then} \]

\[m_{t+1}\]

\[\text{If [response] then} \]

\[m_{t+2}\]

\[\text{else} \]

\[m_4 \leftarrow \text{sign}_O(h(m_3))\]
Contract Signing (Cont’d)

Output: $\text{contract}_R(m_1, m_2)$

Output: $\text{contract}_R(m_1, O)$

- Question:
  - Why can these tokens guarantee NRO or NRR?