Security in Multi–Agent Systems
A Case Study on Comparison Shopping

Dieter Hutter
(joined work with Heiko Mantel, Ina Schaefer, and Axel Schairer)

German Research Center for Artificial Intelligence (DFKI GmbH)
Saarbrücken, Germany
Aspects of Security

Multi-agent security?
Example: (Secure) Comparison Shopping

Matchmaker

List of merchants

Preferences

Offer request

Offer

Buy request

Merchant MA

Customer CA
Example: interests of merchant:

A merchant agent shall not obtain information about the offers made by other merchants to any customer agent.

Goal: Local security requirements to the agents:

- Requirements to customer agent C
  - Offers received from a merchant have to concealed from other merchants
  - ... have to be concealed from other customers
- Requirements to merchant agent M
  - Offers sent to C have to be concealed from other customers
  - ... have to be concealed from other merchant
Agent Modelling

Agent is state–event system with
- state: triple: (Run, pc, mem)
- input events: init, start, recv_a(b, m)
- output events: send_a(b, m)
- internal events: int(pc_1, pc_2, mem)

Platform is a state–event system with
- state: (buf_{a1}, ..., buf_{an})
- input events: send_a(b, m)
- output events: recv_a(b, m)

Specification with the help of pre and postconditions
Example of an agent – the customer

- Initialization (init, start)
- Request for offers
- Collecting offers
- Computing best offer
- Buy request
Formalizing Concealment

Example: Requirement to customer CA

- Offers received from a merchant have to concealed from other merchants

Formalizing confidentiality “to conceal”:

- Messages of CA sent to other agents (except MA) must not depend on offers of MA.

Notice:
- Confidentiality is formalized as notion of dependability
Trace-based system model

Event system: \( ES = (E, I, O, Tr) \)
- \( E \) set of events, e.g. \( recv_a(b, m), send_b(a, m) \)
- \( I, O \subseteq E \) Input/Output events
- \( Tr \subseteq E^* \) set of admissible traces (prefix closed)

View \((V, N, C)\)
- Events are split into confidential \((C)\), visible \((V)\) and non-visible \((N)\) (but not confidential) events (views are local to individual observers)
• Observer has complete knowledge on system behaviour
• Visible events must not depend on secret events
• i.e. set of possible traces (system runs) has to include a trace in which the secret event did not happen. (closure property)

• Basic security predicates
  – Properties on sets of traces (system behaviour!) wrt. a view
  – Closure properties

• Security predicates
  – Conjunction of basic security predicates
• Information flow policies
• Possibilistic models for nondeterministic systems
• System is defined as a set of (acceptable) traces like:

```
• Only events (arrows) are considered!
```
Semantics of Security (Confidentiality)

• Knowing the system (i.e. set of admissible traces)
• Observing only public (green) events (arrows)

Can we deduce anything about possible occurrences of secret (red) events (arrows) ?

Closure property of set of traces!
Security Predicates

- Security Predicates are closure properties on set of traces.

- Intuition: if a trace Tr is an admissible trace in M then there is also some other admissible trace Tr' in M with ....
Definition of visible, non-visible and confidential events for all individual observer $M'$:

**Confidential**: offer of MA to CA

\[\text{send}_{MA}(CA, offer) \quad \text{recv}_{CA}(MA, offer) \quad \text{recv}_{CA}(MA', offer') \quad \text{send}_{MA'}(CA, offer')\]

Formal: definition of a view $\mathcal{V} = (V, N, C)$ as partition of the set of events
Formalizing Confidentiality

1. Keeping the occurrence of an event confidential (view of M’)

   BSD (backward strict deletion)

   $send_{MA}(M,CA,offer)$  $recv_{CA}(MA,offer)$  $recv_{CA}(MA',offer')$

   $send_{MA}(CA,offer')$

2. Keeping the non-occurrence of events confidential

   BSIA (backwards strict insertion of admissible events)

   $send_{MA}(CA,offer)$  $send_{M}(CA,offer)$  $recv_{CA}(MA',offer')$

   $send_{MA}(CA,offer)$  $send_{MA}(CA,offer')$
Uniform representation in MAKS (Mantel)

- security predicate is combined of BSPs
- BSPs prohibit
  - deduction on occurrences of events
  - deduction on non-occurrences of events
- Example: Definition of BSD

\[
\forall \alpha, \beta \in E^*. \forall c \in C. (\beta,\{c\}.\alpha \in Tr \land \alpha_c = \langle \rangle) \\
\Rightarrow \exists \alpha' \in E^*. (\beta.\alpha' \in Tr \land \alpha'_{\cup C} = \alpha_{\cup C})
\]
Composition of Event Systems

Given:
$$ES_1 = (E_1, I_1, O_1, Tr_1) \text{ and } ES_2 = (E_2, I_2, O_2, Tr_2)$$
$$E_1 \cap E_2 \subseteq (O_1 \cap I_2) \cup (O_2 \cap I_1)$$

Then:
$$ES_1 \parallel ES_2 = (E, I, O, Tr) \text{ is defined by:}$$
- $$E = E_1 \cup E_2$$
- $$I = (I_1 \setminus O2) \cup (I_2 \setminus O_1)$$
- $$O = (O_1 \setminus I_2) \cup (O_2 \setminus I_1)$$
- $$Tr = \text{interleaving of traces of } Tr_1 \text{ and } Tr_2$$

Multi-agent system in general: \( ES = (||_{x \in Ag} ES_x) \parallel ES_P \)

Comparison shopping system: \( ES = (||_{x \in CAS} ES_x) \parallel (||_{x \in MAS} ES_x) \parallel ES_P \)
## Composition Preserves (Some) Security Predicates

<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>BSD</th>
<th>BSI</th>
<th>BSIA</th>
<th>FCD</th>
<th>FCI</th>
<th>cond</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BSD</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BSI</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BSIA</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>FCD</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>B,C</td>
<td></td>
</tr>
<tr>
<td>FCI</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>B,D,E</td>
<td></td>
</tr>
</tbody>
</table>
Security requirements define general security property of a multi-agent system

Problem:

how to decompose multi-agent system into a set of subsystems such that:

- each subsystem satisfies appropriate security properties

- composition of the subsystems guarantee general security property.
if

- $ES_{\Phi^+}$ contains all confidential events of the overall system
- there are no N-events in the platform
- $send_a(b, m) \in V_P$ iff $recv_b(a, m) \in V_P$

then $BSD_{ES_{\Phi^+}}$ implies BSD of the entire system
If

• all messages between friends and observers are visible
• for all messages between friends:
  – \( \text{send}_a(b, m) \) is visible iff \( \text{recv}_b(a, m) \) is visible
  – \( \text{send}_a(b, m) \) is confidential iff \( \text{recv}_b(a, m) \) is confidential
• No N–events between platform and friends
• confidential events of the MA–system are also considered as confidential by friends: \( (C \cap E_a \subseteq C_a) \)
• each confidential event of the MA–system is a confidential event of a friend
• for all friends \( a \): BSD holds wrt. \( (E_a, N_a, C_a) \)

then BSD holds for the subsystem of friends \( ES_{\Phi^+} \)
Friends and Observers in Comparison Shopping

- Observer is a merchant MA and all customers except CA
- Friends are the customer CA and all merchants MA‘ except MA

View \((E_{CA}, N_{CA}, C_{CA})\) for customer CA:

- \(C_{CA} = \{ recv_{CA}(MA', o), send_{CA}(MA', Buy(o)) | MA' \neq MA \}\)
- \(N_{CA} = Int_{CA}\)
- \(V_{CA} = E_{CA} \setminus (N_{CA} \cup C_{CA})\)

View \((E_{MA'}, N_{MA'}, C_{MA'})\) for merchant MA‘:

- \(C_{MA'} = \{ recv_{MA'}(CA, buy(o)), send_{MA'}(CA, o) \}\)
- \(N_{MA'} = Int_{MA'}\)
- \(V_{MA'} = E_{MA'} \setminus (N_{MA'} \cup C_{MA'})\)
Proving BSD for Individual Friends

- Unwinding techniques
- Unwinding theorem reduces the verification of information flow properties to more local conditions involving only single transitions

- Unwinding technique
  - Idea: reformulate requirement by local conditions
  - Unwinding conditions: requirements of transitions
  - Theorem: if unwinding conditions hold then BSP holds
Unwinding at a Glance (I)

• State–event systems: \( \text{SES} = (S, s_0, E, I, O, T) \) as before

• Classify states by \textit{guessing} an unwinding relation\(\nless\) 
  \ - \( s \less s' \Rightarrow \) observations possible in \( s \) are possible in \( s' \)
    \ - \( s \less s' \) and \( s' \less s \Rightarrow \) states are indistinguishable

  \ - \( \) suppose \( (s, c, s') \in T \Rightarrow s' \less s \)
    \ - \( \) Observations after confidential event has occurred
      must be possible without this occurrence

• \(\nless\) need not be an equivalence relation
Unwinding Conditions for BSD

∀ \( c \in C \): \( T(s, c, s') \rightarrow s \nless s' \)

∀ \( e \in V \cup N \):
\[
\begin{align*}
&\quad (s_1 \nless s_1' \land T(s_1, e, s_1') \rightarrow \exists s_2' \exists \delta \in (E \setminus C)^* : \\
&\quad T(s_1', \delta, s_2') \land \delta_{|V} = e_V \land s_2 \nless s_2')
\end{align*}
\]
Example:

\[ \text{recv}_{\text{ca}}(\text{ma}, \text{o}) : \text{affects \text{mem}(offers), \text{mem}(OMers), \text{pc}} \]

**Pre:** \( \text{Run} = \text{true}, \text{pc} = 4, \text{ma} \in \text{OMers}, \text{o} \in \text{Offers} \)

**Post:** \( \text{mem}'(\text{Offers}) = \{[\text{ma}, \text{o}]\} \cup \text{mem}(\text{Offers}), \text{mem}'(\text{OMers}) = \text{mem}(\text{OMers}) \setminus \{\text{ma}\}, \text{pc}' = 4 \)

Run = \bot, pc = 0

Run = \bot, pc = 1

Run = T, pc = 1

Run = T, pc = 4

Run = T, pc = 5

Run = T, pc = 6

Run = T, pc = 7

Run = T, pc = 2

Run = T, pc = 3

\[ \text{init}_{\text{ca}}(...) \]

\[ \text{recv}_{\text{ca}}(\text{ma}, \text{o}) \]

\[ \text{int}_{\text{ca}}(\text{1}, \text{4}, \text{Mem}) \]

\[ \text{int}_{\text{ca}}(\text{4}, \text{5}, \text{Mem}) \]

\[ \text{int}_{\text{ca}}(\text{5}, \text{6}, \text{Mem}) \]

\[ \text{int}_{\text{ca}}(\text{1}, \text{2}, \text{Mem}) \]

\[ \text{int}_{\text{ca}}(\text{2}, \text{3}, \text{Mem}) \]

\[ \text{send}_{\text{ca}}(\text{ma}, \text{offer}_{\text{req}}...) \]
Unwinding Relation for Customers

\[
[\text{Run}', \text{pc}', \text{mem}'] \times [\text{Run}', \text{pc}', \text{mem}'] \leftrightarrow \\
\{ \Psi(\text{Run}, \text{pc}) = \Psi(\text{Run}', \text{pc}') \land \\
\text{pc} > 0 \rightarrow \\
( \text{mem}('\text{RMers}) = \text{mem}('\text{RMers}) \land \text{mem}('\text{CMer}) = \text{mem}('\text{CMem}) \\
\land \ \text{MA} \in \text{mem}('\text{OMers}) \leftrightarrow \text{MA} \in \text{mem}('\text{OMers})) \land \\
\land \ \forall \ o \ (\text{MA}, o) = \min(\text{mem}('\text{Offers})) \rightarrow (\text{MA}, o) = \min(\text{mem}(\text{Offers}) \\
\land (\min(\text{mem}('\text{Offers})) \leq \min(\text{mem}(\text{Offers}) ) } \}
\]
Proving Unwinding Properties

∀ c ∈ C: T(s, c, s′) → s \nprecedes\ s′  for c = recv_{ca}(ma′, o), ma' ≠ ma

{ ... pc= 4 \land pc'= 4 \land ma' ∈ OMers \land o ∈ Offers \land ma ≠ ma
\land \text{mem}'(Offers) = \{[ma', o]\} \cup \text{mem}(Offers) \\
\land \text{mem}'(CMer) = \text{mem}(CMer) \land \text{mem}(RMers) = \text{mem}'(RMers) \\
\land \text{mem}'(OMers) = \text{mem}(OMers) \setminus \{ma'\} } \\
→ \\
\{ \Psi(Run, pc) = \Psi(Run', pc') \land \\
\text{pc} > 0 → \\
( \text{mem}(RMers) = \text{mem}'(RMers) \\
\land \text{mem}(CMer) = \text{mem}'(CMer) \\
\land ma ∈ \text{mem}(OMers) \iff ma ∈ \text{mem}'(OMers)) \\
\land \forall o: (ma, o) = \min(\text{mem}'(Offers)) → (ma, o) = \min(\text{mem}(Offers) \\
\land (\min(\text{mem}'(Offers)) \leq \min(\text{mem}(Offers))) \} \}
Proving Unwinding Properties

∀ c ∈ C: T(s, c, s′) → s ⊢ s′ for c = recv_{ca}(ma′, o), ma' ≠ ma

{ ... pc= 4 ∧ pc' = 4 ∧ ma' ∈ OMers ∧ o ∈ Offers ∧ ma ≠ ma
∧ mem'(Offers) = {[ma', o]} ∪ mem(Offers)
∧ mem'(CMer) = mem(CMer) ∧ mem(RMers) = mem'(RMers)
∧ mem'(OMers) = mem(OMers) \ {ma'} }
→
{ Ψ(Run, pc) = Ψ(Run', pc') ∧
  pc > 0 →
  ( True
∧ True
∧ ma ∈ mem(OMers) ↔ ma ∈ mem'(OMers))
∧ ∀ o: (ma, o) = min(mem'(Offers)) → (ma, o) = min(mem(Offers)
∧ (min(mem'(Offers)) ≤ min(mem(Offers)) ) }
Proving Unwinding Properties

\( \forall c \in C: T(s, c, s') \rightarrow s \not\sim s' \) for \( c = \text{recv}_{ca}(ma', o), ma' \neq ma \)

\( \{ \ldots \ pc = 4 \land pc' = 4 \land ma' \in \text{OMers} \land o \in \text{Offers} \land ma \neq ma \) \\
\( \land \text{mem}'(\text{Offers}) = \{[ma', o]\} \cup \text{mem}(\text{Offers}) \) \\
\( \land \text{mem}'(\text{CMer}) = \text{mem}(\text{CMer}) \land \text{mem}(\text{RMers}) = \text{mem}'(\text{RMers}) \) \\
\( \land \text{mem}'(\text{OMers}) = \text{mem}(\text{OMers}) \) \\
\( \rightarrow \) \\
\( \{ \Psi(\text{Run, } pc, \text{cm}), (\text{mem}(\text{pc}), pc) \} \) \\
\( \text{pc} > 0 \rightarrow \) \\
( \( ma \in \text{mem}(\text{OMers}) \leftrightarrow ma \in \text{mem}(\text{OMers}) \setminus \{ma'\} \)) \\
\( \land \forall o: (ma, o) = \text{min}([[ma', o]\} \cup \text{mem}'(\text{Offers})) \) \\
\( \rightarrow (ma, o) = \text{min}(\text{mem}(\text{Offers})) \) \\
\( \land (\text{min}([[ma', o]\} \cup \text{mem}(\text{Offers})) \leq \text{min}(\text{mem}(\text{Offers})) ) \} \)
Future Work and Conclusion

Future Work:
- Automate proof assistance for unwinding proofs
- Decomposition theories for other BSPs

Conclusion:
- Framework for specification and verification of secure (wrt. information flow) multi-agent systems
- Applications in comparison shopping
• Modelling of encrypted messages in possibilistic information flow
• Integrated into the MAKS framework: providing support for composition, refinement, verification
• Allows to detect security problems due to traffic analysis (e.g. verification of mixes)
  – In contrast to using downgrading and intransitive information flow

• Approach assumes perfect encryption
  – Equivalence relation on $E$ is given initially
  – Boot strapping approach (?)