Security in Open Multi-agent Systems

Shahriar Bijani

Feb 2011

Outline

- Open Multi-agent Systems
- Attack Classification
- Probing Attack
- Attack Detection

Outline

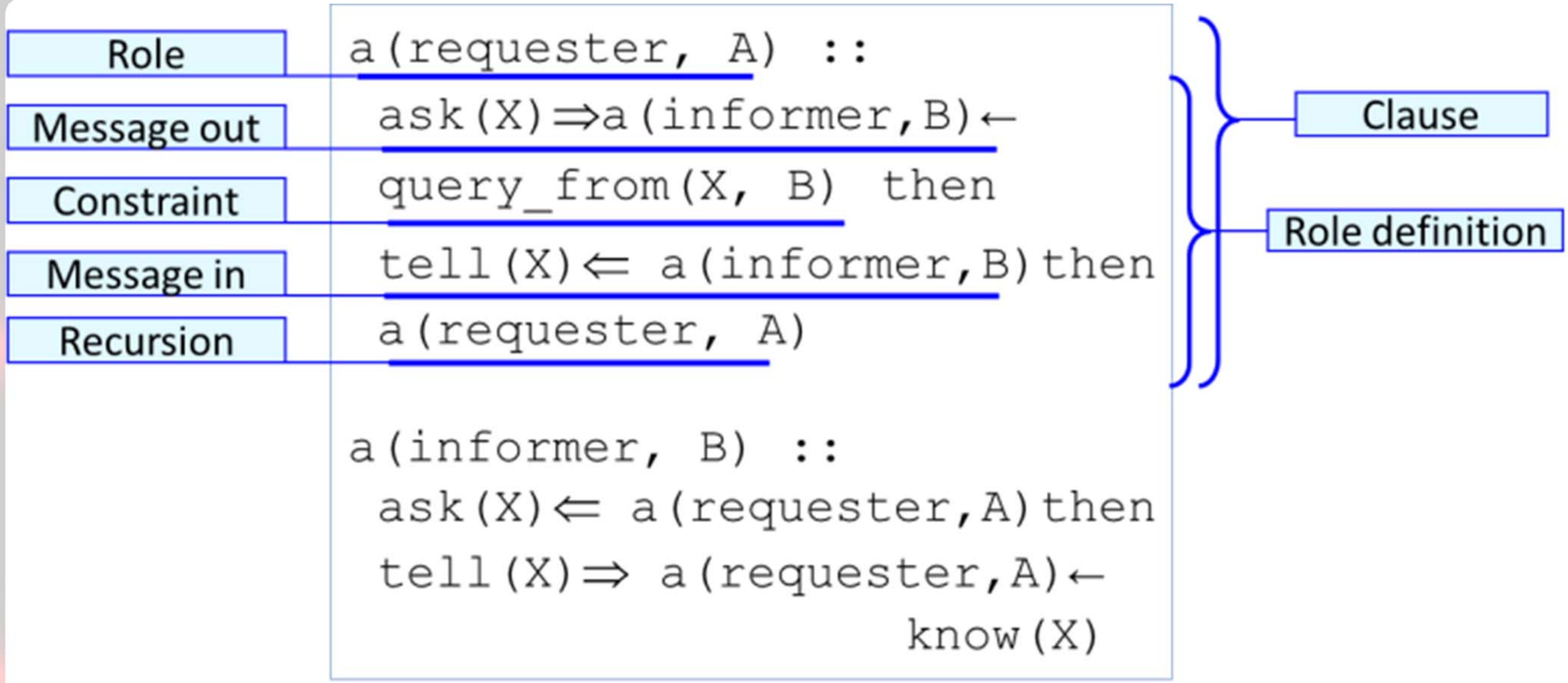
- **Open Multi-agent Systems**
- **Attack Classification**
- **Probing Attack**
- **Attack Detection**

Introduction: Open MAS

- **Open Multi-agent System (MAS)**
  An open system in which autonomous agents can join and leave freely.

- **In Our Analysis:**
  Open peer to peer MAS that agents can invent protocols for different applications and share them.

Introduction: LCC



LCC language syntax:

Outgoing message: ⇒

Incoming message: ⇐

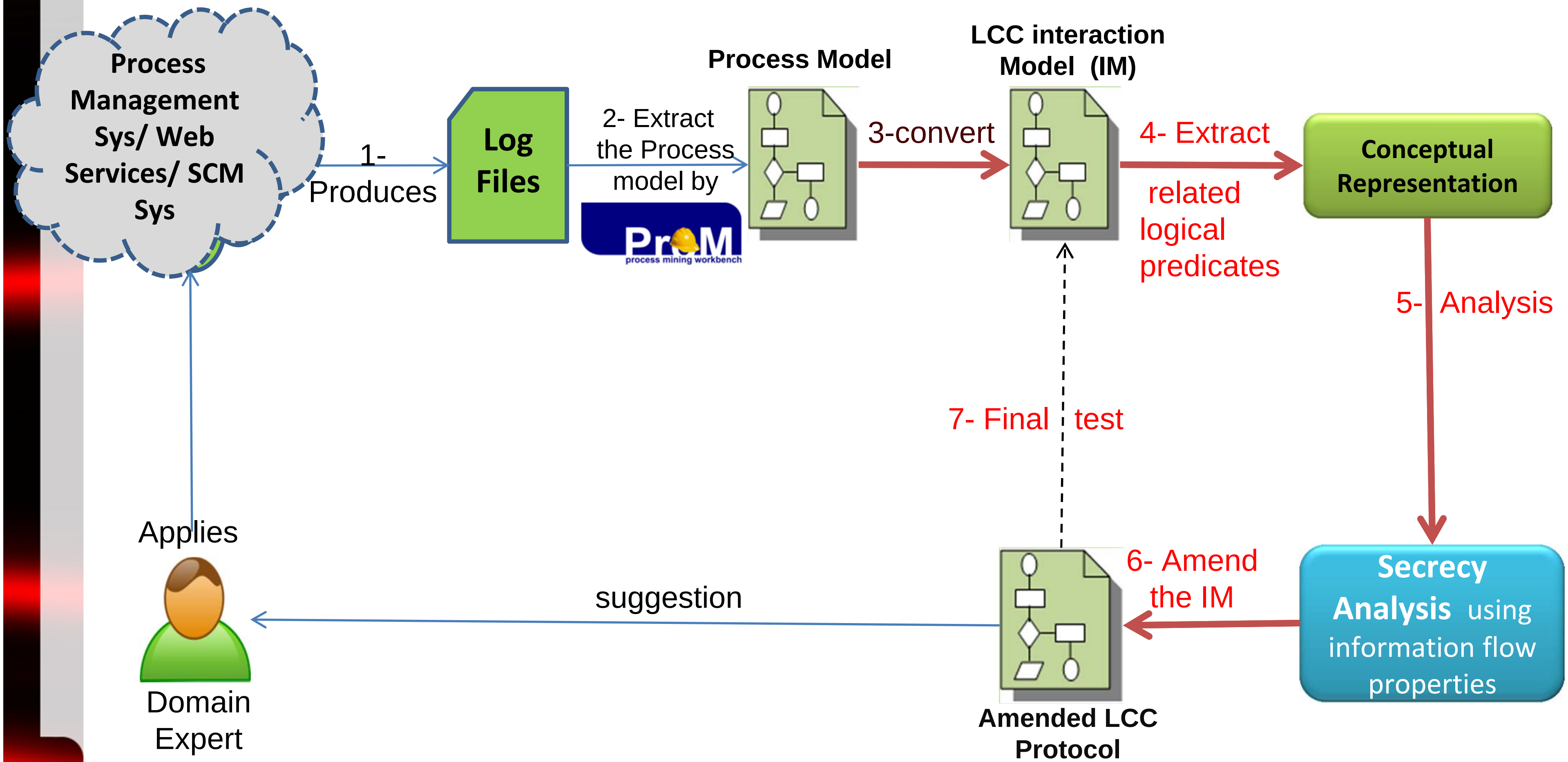
Conditional: ←

Sequence: **then**

Committed choice: **or**

- We might extend the scope of our security analysis without much difficulty to other domains such as web services.

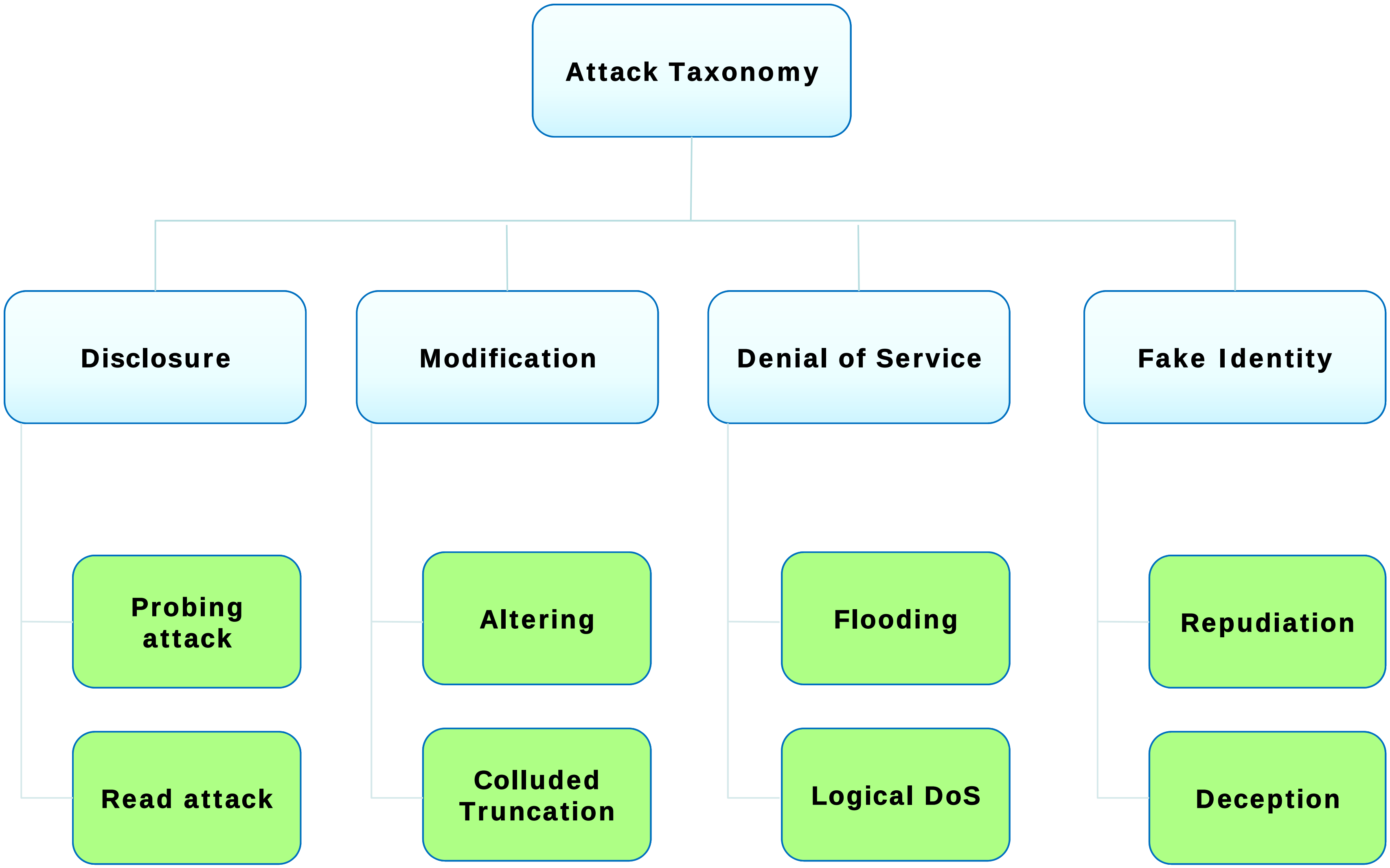
Our Security Analysis Framework



Outline

- Open Multi-agent Systems
- **Attack Classification**
- Probing Attack
- Attack Detection

Attack Classification



Attacks on Open MAS

Disclosure	-To intercept data, code, protocols, … -Tracing the interaction models
Modification	-To alter transferring information -Malicious agents collude to modify
Fake Identity	-An attacker plays many pseudonymous roles -A victim may enter into a fake interaction -A method to deceive the *trust service*
Denial of Service	-To prevent agents to use provided service -To waste others' resources -To ruin the reputation

Outline

- **Open Multi-agent**
- **Attack Classification**
- **Probing Attack**
- **Attack Detection**

Probing Attack

- an adversary could infer information not only from the interaction model itself, but also from the local knowledge of other agents.

- Four types of probing attacks:
  1. explicit query
  2. implicit query
  3. injection attack
  4. indirect query

Implicit Probing Attack

An Example of Proteomics Lab Interaction Model

```
a(researcher(LabList), R) ::
   ( ask(X)=> a(omicslab, H) <-
                LabList=[H|T] then
     tell(Y)<= a(omicslab,H) then
     null <- processResult(X,Y,H)
     then a( researcher(T), R)
    ) or
    null <- LabList = []

a(omicslab, O) ::
   ask(X)<= a(researcher,R) then
   tell(Y)=>a(researcher,R)<-
                              Combine(X,Y)
 then a(omicslab, O)
```

Injection Attack

Injection Attack

- A Selling Interaction Model:

```
a(vendor, V)::
 null <- ask(S)<= a(customer,C) then
 null <- (not(want(C,S)) or  payFor(C,S) ) then   /*injection */
 null <- (not(SupplyFrom(X)) or want(C,S) ) then /* injection */
 ok => a(customer,C) <- agree(C,S)     then  /* implicit query */
  ...
 then a(vendor, V)
```

Outline

- Open Multi-agent Systems
- Attack Classification
- Probing Attack
- **Attack Detection**

Injection Attack

- A Selling Interaction Model:

```
a(vendor, V)::
 null <- ask(S)<= a(customer,C) then
 null <- (not(want(C,S)) or  payFor(C,S) ) then   /*injection */
 null <- (not(SupplyFrom(X)) or want(C,S) ) then /* injection */
 ok => a(customer,C) <- agree(C,S)     then  /* implicit query */
  ...
 then a(vendor, V)
```

- Conceptual representation:

  IM_1 ={

  want(C,S) → payFor(C,S),

  SupplyFrom(X) → want(C,S)

  }

  q_1 ={**agree(C,S)**}

Query result: **send(C)**

Attack Detection

- We formulate the problem as $A_0 \cup IM_i \vdash q$

  Where:

  A_0 = Agent's Local Knowledge

  IM_i = The injection parts of an Interaction Model

  q = query

- Example: $A_0 \cup IM_1 \vdash q_1$

  $A_0 \cup IM_2 \vdash q_1$

  $A_0 \cup IM_3 \vdash q_1$

Attack Detection

- **Detectability** (or non-opacity): an information flow property that shows the ability to infer a specific predicate from a set of rules

- We use an inference system for *detectability* [Becker 2010]

$$\text{(PEEK)}\ \frac{IM \vdash q \quad q\ is\ monotonic, \quad IM\ is\ ground}{\Pi, IM \Vdash q}$$

$$(WEAK)\ \frac{\Pi, IM \Vdash q \qquad q \Rightarrow q'}{\Pi, IM \Vdash q'}$$

$$\text{(POKE1)}\ \frac{A_0 \cup IM \vdash q \qquad (IM, q) \in P}{\Pi, IM \Vdash q}$$

$$\text{(POKE2)}\ \frac{A_0 \cup IM \nvdash q \qquad (IM, q) \in P}{\Pi, IM \Vdash \neg q}$$

$$\text{(MONO1)}\ \frac{\Pi, IM \Vdash q \qquad IM' \succcurlyeq IM \quad q\ is\ monotonic \quad IM'\ is\ ground}{\Pi, IM' \Vdash q}$$

$$\text{(MONO2)}\ \frac{\Pi, IM \Vdash q \qquad IM' \preccurlyeq IM \quad \neg q\ is\ monotonic \quad IM'\ is\ ground}{\Pi, IM' \Vdash q}$$

$$\text{(CONJ)}\ \frac{\Pi, IM \Vdash q_1 \qquad \Pi, IM \Vdash q_2}{\Pi, IM \Vdash q_1 \wedge q_2}$$

$$\text{(DIFF)}\ \frac{\Pi,\ A_0 \cup IM \Vdash q}{\Pi, A_0 \Vdash q \vee \bigvee_{a \in IM} fired(IM, head(a))}$$

Countermeasures

- Two reasons that security problems lead to probing attacks:

  (1) no distinguishing notion of private and public data in LCC

  (2) no mechanism for information leakage control in an interaction.

Countermeasures

- Prevention

  change the syntax of LCC

- Detection

  An intrusion detection interaction model to interpret other interaction models