May 3: Trust and Hybrid Models

• Trust models

• Chinese Wall model
  – Aggressive Chinese Wall model
Types of Trust Models

- Policy-based trust management
- Recommendation-based trust management
Policy-Based Trust Management

• Policy *rules* determine whether to trust
• Credentials provide instantiation information
  – Credentials themselves may be input to rules
  – Trusted third parties may be involved
• Generally assume agents act autonomously
Keynote

• Rule-based trust management system
• Policy assertions: statements about policy
• Credential assertions: describe actions allowed by credentials
• Action environment: set of attributes describing action associated with set of credentials
Evaluator

• Inputs
  – Policy assertions describing local policy
  – Set of credentials
  – Action environment

• Applies instantiated assertions to action environment

• Outputs
  – Whether proposed action consistent with local policy
Example: Email Domain

Policy, credential assertions:

Local-Constants: Alice="cred1234", Bob="credABCD"
Authorization: "authcred"
Licensees: Alice || Bob
Conditions: (app_domain == "RFC822-EMAIL") &&
            (address ~= "^.*@keynote\.ucdavis\.edu$")
Signature: "signed"

entity with “authcred” credentials trust holders of “cred1234”,
“credABCD” to issue credentials (“signed”) for users in email
domain when address ends in “@keynote.ucdavis.edu"
Example: Email Domain

Compliance values: _MAX_TRUST, _MIN_TRUST

Action environment:

_ACTION_AUTHORIZERS=Alice
app_domain = "RFC822-EMAIL"
address = "opus@keynote.ucdavis.edu"

Satisfied; output _MAX_TRUST
Example: Separation of Duty

Invoicing system delegates authority for payment of invoices to entity with credential fundmgrcred

Policy assertion:

Authorizer: "POLICY"
Licensee: "fundmgrcred"
Conditions: (app_domain == "INVOICE" &&

@dollars < 10000)
Example: Separation of Duty

Credential assertion requiring at least 2 signatures on expenditure:

Comment: specifies a spending policy
Authorizer: "authcred"
Licensees: 2-of("cred1", "cred2", "cred3",
               "cred4", "cred5")

Conditions: (app_domain=="INVOICE")
    -> { (@dollars) < 2500) -> _MAX_TRUST;
       (@dollars < 7500) -> "ApproveAndLog"; }
Signature: "signed"
Example: Separation of Duty

Compliance values: Reject, ApproveAndLog, Approve

Action environment:

_ACTIONAUTHORIZEDS = "cred1,cred4"
app_domain = "INVOICE"
dollars = "1000"

Satisfied; output Approve
Example: Separation of Duty

Action environment:

_DATABASE_COLUMN_1 = "cred1,cred2"
app_domain = "INVOICE"
dollars = "3541"

Satisfied; output ApproveAndLog
Example: Separation of Duty

Action environment:

.getActionAuthorizers = "cred1"
app_domain = "INVOICE"
dollars = "1500"

.getActionAuthorizers = "cred1,cred5"
app_domain = "INVOICE"
dollars = "8000"

Not satisfied; output Reject
Reputation-Based Trust Management

• Trust based on past behavior, especially during interactions, and other information
  – May include other recommendations
  – Each entity maintains its own list of relationships
Types of Trust

• Direct trust
  – Amy trusts Boris

• Recommender trust
  – Amy trusts Boris to make recommendations about others
Example: Abdul-Rahman, Hailes

- **Trust value semantics**

<table>
<thead>
<tr>
<th>value</th>
<th>DT meaning</th>
<th>RT meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>–1</td>
<td>Untrustworthy</td>
<td>Untrustworthy</td>
</tr>
<tr>
<td>0</td>
<td>Cannot make trust judgment</td>
<td>Cannot make trust judgment</td>
</tr>
<tr>
<td>1</td>
<td>Lowest trust level</td>
<td>*</td>
</tr>
<tr>
<td>2</td>
<td>Average trustworthiness</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>More trustworthy than most entities</td>
<td>*</td>
</tr>
<tr>
<td>4</td>
<td>Completely trustworthy</td>
<td>*</td>
</tr>
</tbody>
</table>
Example

- Amy needs Boris’ recommendation about Danny
  - Amy trusts Boris recommendation with value 2
- Boris doesn’t know Danny, so asks Carole
- Carole replies with recommendation of 3
- Boris adds his name to recommendation, sends it on
Amy’s Computation

• 4 entities involved: Amy, Boris, Carole, Danny
• $tv(Amy:Boris)/4 \times tv(Boris:Carole)/4 \times tv(Carole:Danny)/4 =\frac{2}{4} \times \frac{3}{4} \times 3 = \frac{9}{8}$
Main Issue

• How do you populate the initial matrix
  – That is, how do you set the trust values for each pair of entities
Example: PeerTrust

- Based on complaints as feedback
  - $P$ peer-to-peer network, $u$ node
  - $p(u, t)$ node that $u$ interacts with in transaction $t$
  - $S(u, t)$ amount of satisfaction $u$ gets from $p(u, t)$
  - $I(u)$ total number of transactions $u$ does
  - $Cr(v)$ credibility of node $v$’s feedback
Example: PeerTrust

• Trust value of $u$ is:

$$T(u) = \sum_{t=1}^{I(u)} S(u, t) Cr(p(u, t))$$

• where $Cr(v)$ is (one of many possible):

$$Cr(v) = \sum_{t=1}^{I(v)} S(v, i) \frac{T(p(v, t))}{\sum_{x=1}^{I(v)} I(v) T(p(v, x))}$$
Key Points

• Integrity policies deal with trust
  – As trust is hard to quantify, these policies are hard to evaluate completely
  – Look for assumptions and trusted users to find possible weak points in their implementation
• Biba, Lipner based on multilevel integrity
• Clark-Wilson focuses on separation of duty and transactions
Chinese Wall Model

Problem:

– Tony advises American Bank about investments
– He is asked to advise Toyland Bank about investments

• Conflict of interest to accept, because his advice for either bank would affect his advice to the other bank
Organization

- Organize entities into “conflict of interest” classes
- Control subject accesses to each class
- Control writing to all classes to ensure information is not passed along in violation of rules
- Allow sanitized data to be viewed by everyone
Definitions

• **Objects**: items of information related to a company
• **Company dataset (CD)**: contains objects related to a single company
  – Written $CD(O)$
• **Conflict of interest class (COI)**: contains datasets of companies in competition
  – Written $COI(O)$
  – Assume: each object belongs to exactly one $COI$ class
Example

- Bank COI Class:
  - Bank of America
  - Citibank
  - Bank of the West

- Gasoline Company COI Class:
  - Shell Oil
  - Standard Oil
  - Union ‘76
  - ARCO
Temporal Element

• If Anthony reads any CD in a COI, he can never read another CD in that COI
  – Possible that information learned earlier may allow him to make decisions later
  – Let $PR(S)$ be set of objects that $S$ has already read
CW-Simple Security Condition

- $s$ can read $o$ iff either condition holds:
  1. There is an $o'$ such that $s$ has accessed $o'$ and $CD(o') = CD(o)$
     - Meaning $s$ has read something in $o'$'s dataset
  2. For all $o' \in O$, $o' \in PR(s) \Rightarrow COI(o') \neq COI(o)$
     - Meaning $s$ has not read any objects in $o$'s conflict of interest class

- Ignores sanitized data (see below)

- Initially, $PR(s) = \emptyset$, so initial read request granted
Sanitization

- Public information may belong to a CD
  - As is publicly available, no conflicts of interest arise
  - So, should not affect ability of analysts to read
  - Typically, all sensitive data removed from such information before it is released publicly (called sanitization)

- Add third condition to CW-Simple Security Condition:
  3. \( o \) is a sanitized object
Writing

- Anthony, Susan work in same trading house
- Anthony can read Bank 1’s CD, Gas’ CD
- Susan can read Bank 2’s CD, Gas’ CD
- If Anthony could write to Gas’ CD, Susan can read it
  - Hence, indirectly, she can read information from Bank 1’s CD, a clear conflict of interest
CW-*-Property

- *s* can write to *o* iff both of the following hold:
  1. The CW-simple security condition permits *s* to read *o*; and
  2. For all unsanitized objects *o'*, if *s* can read *o'*, then $CD(o') = CD(o)$

- Says that *s* can write to an object if all the (unsanitized) objects it can read are in the same dataset
Formalism

- Goal: figure out how information flows around system
- $S$ set of subjects, $O$ set of objects, $L = C \times D$ set of labels
- $l_1: O \rightarrow C$ maps objects to their COI classes
- $l_2: O \rightarrow D$ maps objects to their CDs
- $H(s, o)$ true iff $s$ has or had read access to $o$
- $R(s, o)$: $s$’s request to read $o$
Axioms

• Axiom 7-1. For all \( o, o' \in O \),
  if \( l_2(o) = l_2(o') \), then \( l_1(o) = l_1(o') \)
  – CDs do not span COIs.

• Axiom 7-2. \( s \in S \) can read \( o \in O \) iff,
  for all \( o' \in O \) such that \( H(s, o') \), either
  \( l_1(o') \neq l_1(o) \) or \( l_2(o') = l_2(o) \)
  – \( s \) can read \( o \) iff \( o \) is either in a different COI
    than every other \( o' \) that \( s \) has read, or in the
    same CD as \( o \).
More Axioms

• Axiom 7-3. \( \neg H(s, o) \) for all \( s \in S \) and \( o \in O \) is an initially secure state
  – Description of the initial state, assumed secure

• Axiom 7-4. If for some \( s \in S \) and all \( o \in O \), \( \neg H(s, o) \), then any request \( R(s, o) \) is granted
  – If \( s \) has read no object, it can read any object
Which Objects Can Be Read?

- Suppose $s \in S$ has read $o \in O$. If $s$ can read $o' \in O$, $o' \neq o$, then $l_1(o') \neq l_1(o)$ or $l_2(o') = l_2(o)$.
  - Says $s$ can read only the objects in a single CD within any COI.
Proof

Assume false. Then
\[ H(s, o) \land H(s, o') \land l_1(o') = l_1(o) \land l_2(o') \neq l_2(o) \]

Assume \(s\) read \(o\) first. Then \(H(s, o)\) when \(s\) read \(o\), so by Axiom 7-2, either \(l_1(o') \neq l_1(o)\) or \(l_2(o') = l_2(o)\), so
\[(l_1(o') \neq l_1(o) \lor l_2(o') = l_2(o)) \land (l_1(o') = l_1(o) \land l_2(o') \neq l_2(o))\]

Rearranging terms,
\[(l_1(o') \neq l_1(o) \land l_2(o') \neq l_2(o) \land l_1(o') = l_1(o)) \lor (l_2(o') = l_2(o) \land l_2(o') \neq l_2(o) \land l_1(o') = l_1(o))\]

which is obviously false, contradiction.
Lemma

• Suppose a subject \( s \in S \) can read an object \( o \in O \). Then \( s \) can read no \( o' \) for which 
  \( l_1(o') = l_1(o) \) and \( l_2(o') \neq l_2(o) \).
  
  – So a subject can access at most one CD in each COI class
  
  – Sketch of proof: Initial case follows from Axioms 7-3, 7-4. If \( o' \neq o \), theorem immediately gives lemma.
COIs and Subjects

• Theorem: Let \( c \in C \) and \( d \in D \). Suppose there are \( n \) objects \( o_i \in O, 1 \leq i \leq n \), such that \( l_1(o_i) = d \) for \( 1 \leq i \leq n \), and \( l_2(o_i) \neq l_2(o_j) \), for \( 1 \leq i, j \leq n, i \neq j \). Then for all such \( o \), there is an \( s \in S \) that can read \( o \) iff \( n \leq |S| \).
  
  – If a COI has \( n \) CDs, you need at least \( n \) subjects to access every object
  
  – Proof sketch: If \( s \) can read \( o \), it cannot read any \( o' \) in another CD in that COI (Axiom 7-2). As there are \( n \) such CDs, there must be at least \( n \) subjects to meet the conditions of the theorem.