Supporting Structured Credentials and Sen sitive Policies through Interoperable Strateg ies for Automated Trust Negotiation

Mohammad Hammoud

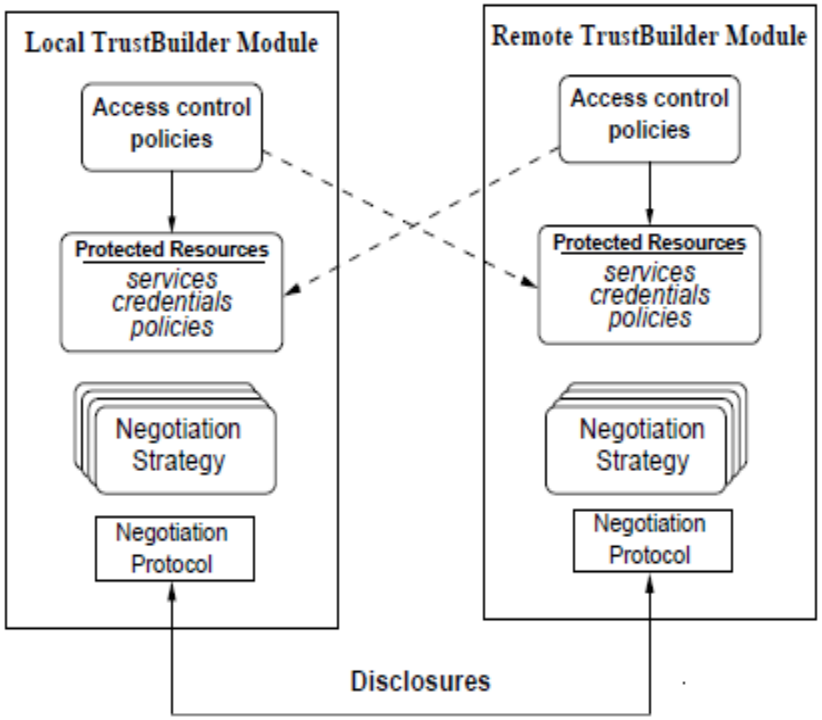
CS3525

Dept. of Computer Science
University of Pittsburgh

Automated (Distributed) Trust Negotiation

- The process of establishing trust between parties (strangers) so that interactions can take place is known as ***automated trust negotiation***.
- Access control polices are used to specify what combinations of digital credentials a stranger must disclose to gain access to a local resource.
- To preserve parties' autonomy:
  - A party should be able to choose its negotiation strategy independently.
  - Parties' strategies need to interoperate.

TrustBuilder: A Prototype System fo r Trust Negotiation



The TrustBuilder Architecture

- Trust is established "incrementally" by exchanging credentials and requests for credentials → ***Trust Negotiation***
- The ordering of messages and the type of information messages will contain is referred to as → ***Trust Negotiation Protocol***
- Controlling the exact content of the messages (i.e., which credentials to disclose next, when to disclose them, when to terminate a negotiation, and accepting new disclosures from the other party) is referred to as → ***Trust Negotiation Strategy***
- Disclosure of protected resources is governed by access control policies, which specify what credentials the other party needs to disclose in order to gain access to local resources.

The Proposed Work

- ***Key Idea:*** a unique underlying protocol is defined so that different strategies can interoperate: they design a strategy-independent, language-independent trust negotiation protocols (1 & 2) that insure the interoperability of different strategies within the TrustBuilder trust negotiation architecture.

- They characterize a broad class of trust negotiation strategies (DTS & BTS).

- This is done for:
  - Propositional credentials and unprotected access control policies.

  - Credentials with internal structure and access control policies whose contents are sensitive.

Talk roadmap

- Black-box credentials:
  - Definitions.
  - A simple example.
  - TrustBuilder Protocol 1.
  - Characterizing safe disclosure sequences.
  - The Disclosure Tree Strategy (DTS) family.
- Credentials with internal structure and sensitive policies:
  - Policy graphs.
  - Bindings.
  - TrustBuilder Protocol 2.
  - Characterizing safe disclosure sequences.
  - The Binding Tree Strategy (BTS) family.

Definitions

- ***Resource*** refers to a credential or service.

- Each resource has exactly one access control policy *P(C1, ….Ck),* where *P(C1, ….Ck)* is a Boolean expression involving only credentials C_1, …, C_k that the other party may possess, Bo olean constants *true* and *false*, the Boolean operators ∨ and ∧, and parentheses as needed.

- Resource R is ***unlocke d*** if its access control policy is satisfied by the set of credentials disclosed by the other party.

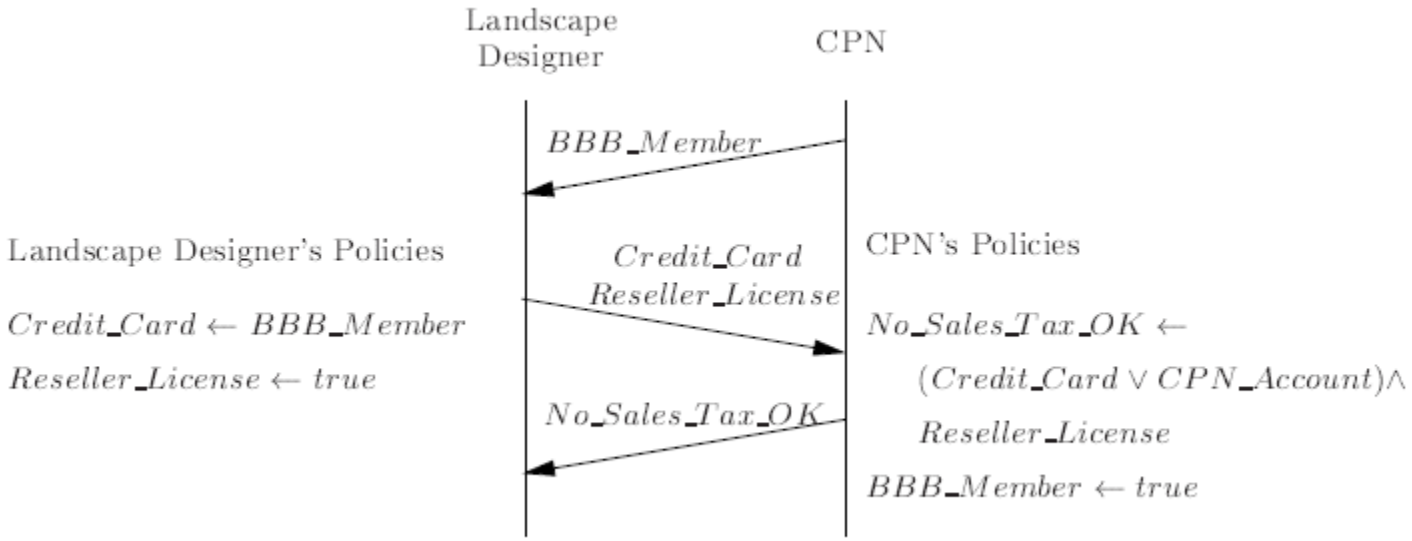
- A resource is ***unprotected*** if its policy is always satisfied.

- The ***denial policy*** C ← *false* mean s that either the party doesn't posses C, or else will not disclose C under any circumstances.

- Given a sequence Q = (C_1, …, C_n) of disclosures of protected resources, if each C_i is unl o cked at the time it is disclosed, $1 \leq i \leq n$, then Q is said to be a ***safe disclosure sequence***.

A Simple Example

Suppose that Alice is a landscape designer who wishes to order plants from Champaign Paririe Nursery (CPN). She fills out an order form on the web , checking an order form box to indicate that she wishes to be exempt from sales tax. Upon receipt of the order, CPN will want to see a valid credit card or Alice's account credential issued by CPN, and a current reseller's license. Alice has no account with CPN, but she does have a digital credit card. She is willing to show her reseller's license to anyone, but she will only show her credit card to members of the Better Business Bureau.



The TrustBuilder Protocol 1

1. The client sends the original resource request message to the server indicating its desire to access service R.

2. The request triggers the negotiation, and the server invokes its local security agent.

3. Meanwhile, the client also invokes its security agent.

4. The client and the server then exchange messages until either the service R is disclosed by the server or one party sends a failure message.

The TrustBuilder Protocol 1: Pseudocode

```
TrustBuilder_Agent(L,R)
Input: L is the set of local resources and policies.
       R is the resource to which the client originally requested access.
Output: the result of a negotiation, which can either be FAIL or SUCCEED.
Let M be an empty disclosure message sequence.
r=NOT_TERMINATED.
If (R is a local resource) then //Negotiation is initiated by the other party.
    r=TrustBuilder_send_response(M,L,R).
    If (r==SUCCEED or r==FAIL) then //Negotiations have terminated.
        return r.
While (r==NOT_TERMINATED)
    Receive message m from the other party.
    Add m to the end of M.
    r=TrustBuilder_check_for_termination(m,R).
    If (r==SUCCEED or r==FAIL) then return r.
    r=TrustBuilder_send_response(M,L,R).
    If (r==SUCCEED or r==FAIL) then return r.
End of TrustBuilder_Agent.

TrustBuilder_send_response(M,L,R)
S_m=Local_strategy(M,L,R).
//S_m contains the candidate messages the local strategy suggests.
Choose any single message m' from S_m.
Send m' to the remote party.
Add m' to the end of M.
r=TrustBuilder_check_termination(m',R).
return r.
End of TrustBuilder_send_response.

TrustBuilder_check_for_termination(m, R)
If (m == ∅) then return FAIL. //Negotiations have failed.
If (R ∈ m) then return SUCCEED. //Negotiations have succeeded.
return NOT_TERMINATED.
End of TrustBuilder_check_for_termination.
```

The TrustBuilder Protocol 1: Definitions

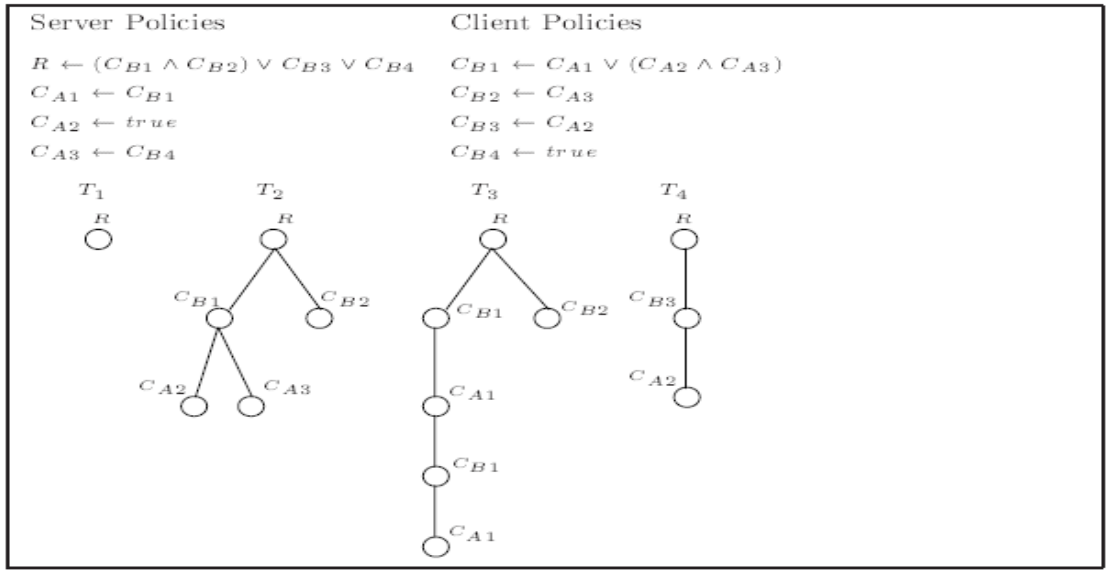
DEFINITION 4.1. A strategy is a function *f* which takes three parameters M, L, and R, where R is the resource to which the client originally requested access, M = (m_1, …, m_k) is a sequence of disclosure messages such that $m_i \neq \emptyset$ and R doesn't belong m_i for $1 \leq i \leq k$, and L is the set of local resources and policies. The output of *f* is S_m, which is a set of disclosure messages.

DEFINITION 4.2. Strategies f_A and f_B are compatible if whenever there exists a safe disclosure sequence for a party Alice to obtain access to a resource owned by party Bob, the trust negotiation will succeed when Alice uses f_A and Bob uses f_B. If $f_A = f_B$, then f_A is said to be self-compatible.

DEFINITION 4.3. A strategy family is a set *F* of mutually compatible strategies, i.e., for every f_1 belong *F*, f_2 belong *F*, f_1 and f_2 are compatible. A set *F* of strategies is said to be closed if given a strategy f', if f' is compatible with every strategy in *F*, then f' belong *F*.

Characterizing Safe Disclosure Sequences

- Trees and tree operations are defined to give meaning to negotiation strategies' actions.
- Negotiation strategies do not need to materialize these trees; rather, the trees provide the formal basis for what a strategy does.



When all the leaves of a disclosure tree T are unprotected credentials, T is said to be a ***full disclosure tree****. Given a disclosure tree T, if there is a credential appearing twice in the path from a leaf node to the root, then T is called a* ***redundant disclosure tree****.*

Characterizing Safe Disclosure Sequences

- **THEOREM 5.1.** Given a non-redundant safe disclosure sequence Q = (C1, …. Cn = R), there is a full non-redundant disclosure tree T such that the following both hold:
  1) The nodes of T are a subset of {C1, …., Cn}.
  2) For all credential pairs (C1, C2) such that C1 is an ancestor of C2 in T, C2 is disclosed before C1 in Q.

- **THEOREM 5.2.** Given a full disclosure tree for R, there is a non-redundant safe disclosure sequence ending with the disclosure of R.

- **COROLLARY 5.1.** There is a safe disclosure sequence ending with the disclosure of R if and only if there is a full non-redundant disclosure tree.

Characterizing Safe Disclosure Sequences

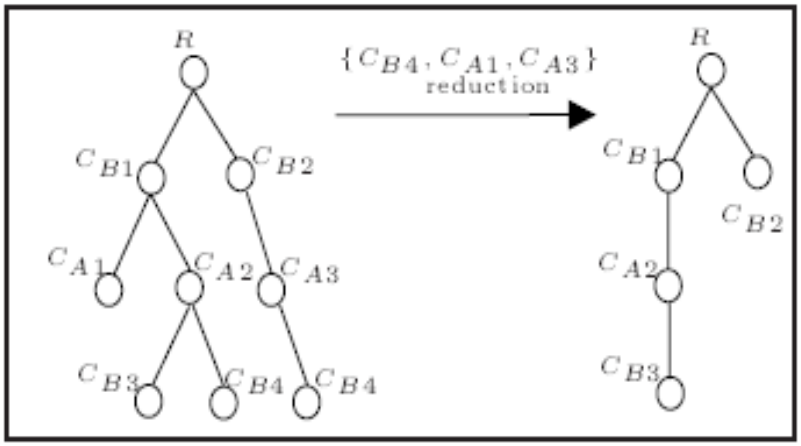
- **Key Point:**

Since there is a natural mapping between safe disclosure sequences and disclosure trees, during the negotiation, theoretically one could determine whether a potential credential or policy disclosure is helpful by examining all the disclosure trees for R

How can this be achieved??

Characterizing Safe Disclosure Sequences

DEFINITION 5.2. Given a disclosure tree T and a set C of credentials, the reduction of T by C, *reduction*(T, C*)*, is the disclosure tree T' which is obtained by removing all the subtrees rooted at a node labeled with resource $C \in$ C. Given a set *T* of disclosure trees, *reduction(T,* C*)* = {*reduction(*T, C) | T $\in$ *T*}.



Characterizing Safe Disclosure Sequences

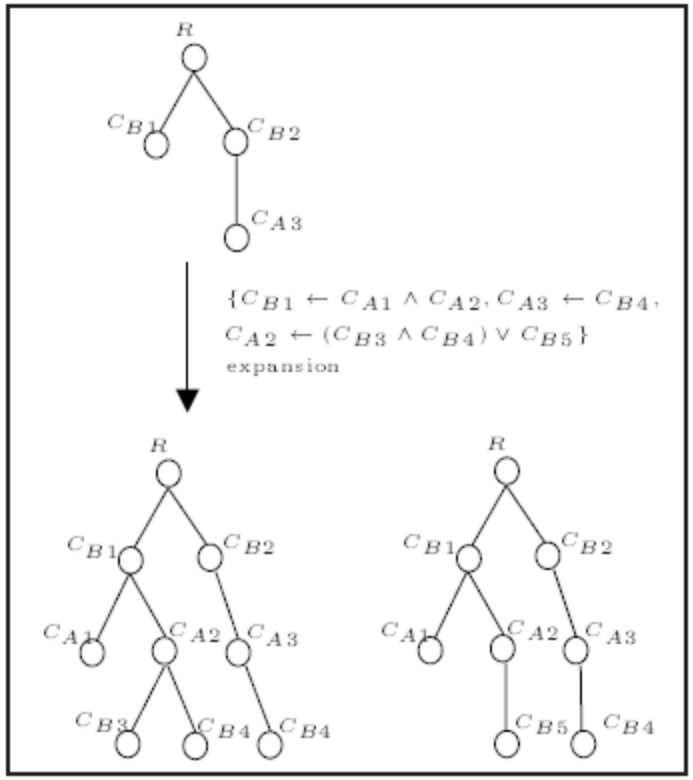
DEFINITION 5.3. Given a disclosure tree T and a policy set P containing no denial policies, the expansion of T by P, *expansion(*T, P*)*, is the set of all disclosure trees T_i such that:

1) T_i can be reduced to T, i.e., there exists a set C of credentials such that reduction(T_i, C) = T.
2) For each edge (C_1, C_2) in T_i, if (C_1, C_2) is not an edge of T, then C_1's policy is in P.
3) For each leaf node C of T_i, either P does not contain C's policy, or T_i is redundant.

Given a set of disclosure trees *T*, *expansion(T,* P)= $\cup_{T \in T}$ *expansion*(T,P)

Characterizing Safe Disclosure Sequences

DEFINITION 5.3. **Example:**



Characterizing Safe Disclosure Sequences

DEFINITION 5.4. Given a set *T* of disclosure trees and a set P_d of denial policies, the denial pruning of *T* by P_d, denoted $prune_{denial}(T, P_d)$, is the set:

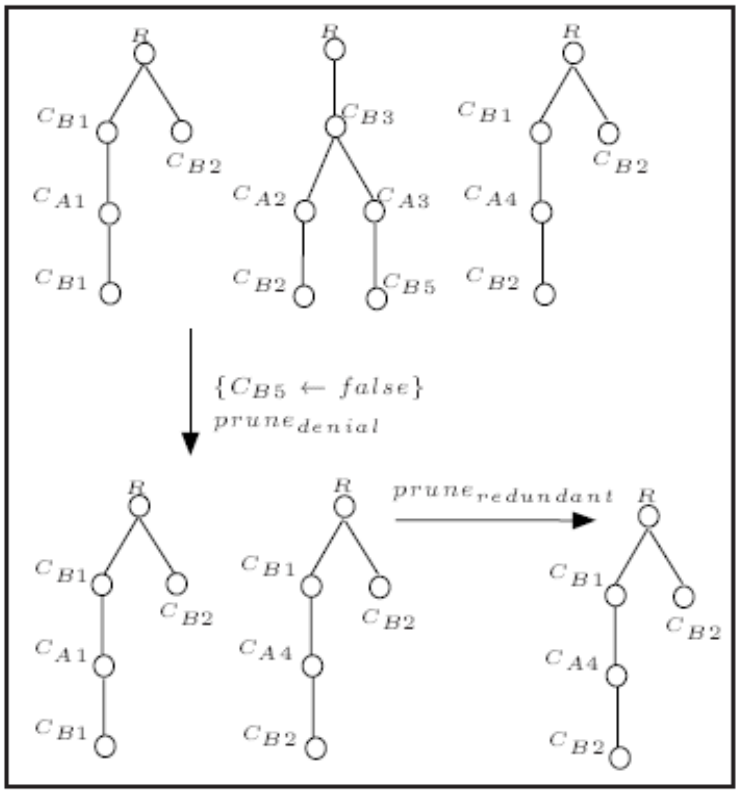
$$\{T \mid \mathrm{T} \in T \text{ and } \mathrm{T} \text{ contains no resource whose policy is in } \mathrm{P_d}\}$$

DEFINITION 5.5. Given a set *T* of disclosure, the redundancy pruning of *T*, denoted $prune_{redundant}(T)$, is the set:

$$\{T \mid \mathrm{T} \in T \text{ and } \mathrm{T} \text{ is not a redundant disclosure tree}\}$$

Characterizing Safe Disclosure Sequences

DEFINITIONS 5.4 & 5.5. **Example:**



Characterizing Safe Disclosure Sequences

DEFINITION 5.6. Given a disclosure tree T and a set P_d of denial policies, a set *P* of non-denial policies, and a set C of credentials, let S = P_d U *P* U C. The evolution of T by S, denoted *evolution(*T, S*)*, is:

$$prune_{redundant}(\, prune_{denial}(reduction(expansion(\mathrm{T}, P), \mathrm{C}), \mathrm{P_d})$$

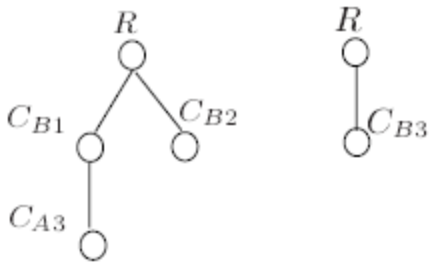
During the negotiation, let S be the set of credentials and policies disclosed so far and L be the local policies of a negotiation party. The *veiw(*S U L*)* contains all the relevant disclosure trees which can be seen by this party.

Characterizing Safe Disclosure Sequences

DEFINITION 5.6. **Example:**

$S = \{C_{A2}, C_{B4},$
$R \leftarrow (C_{B1} \wedge C_{B2}) \vee (C_{B3} \wedge C_{B4}) \vee C_{B5},$
$C_{B1} \leftarrow C_{A1} \vee (C_{A2} \wedge C_{A3}),$
$C_{B3} \leftarrow C_{A2}, C_{B5} \leftarrow C_{A6} \wedge C_{A7},$
$C_{A1} \leftarrow C_{B1}, C_{A3} \leftarrow C_{B4}, C_{A6} \leftarrow false\}$

$view(S)$ contains the following derivation trees:



Answer to the key point question: **from the left disclosure tree, for instance, party B can realize that credential C_{B2} must be disclosed if the negotiation is to succeed. Also this tree is said to be *evolvable* for party B.**

Characterizing Safe Disclosure Sequences

- Although disclosure trees are a useful tool for understanding strategy properties, it would require exponential time and space to materialize all the disclosure trees during a negotiation.

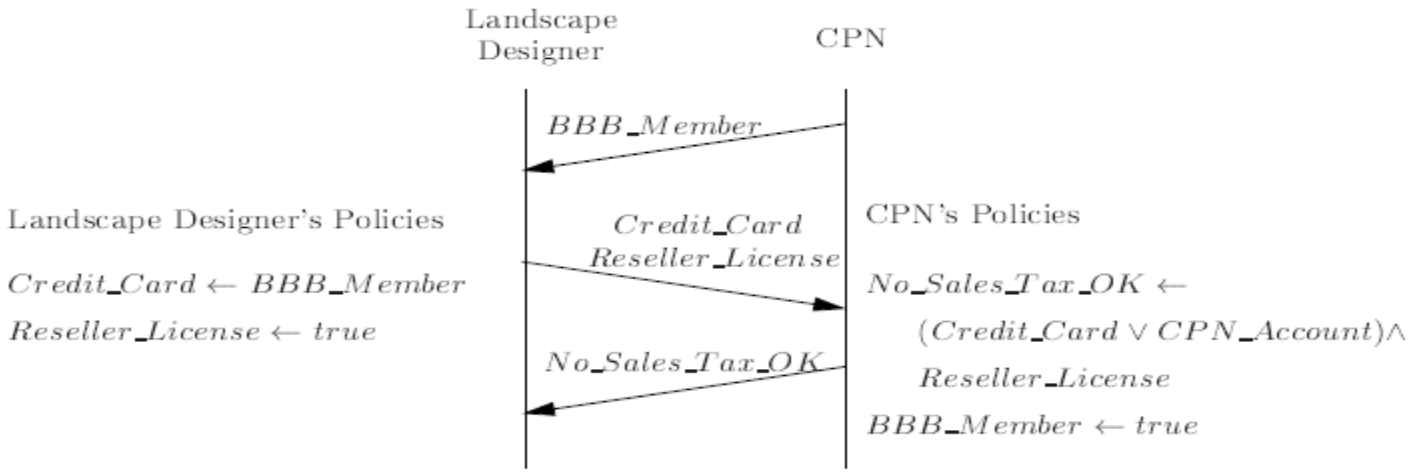
- **TrustBuilder1-Simple**: put all undisclosed policies and unlocked credentials in the next message to the other party. If all the policies and unlocked credentials have already been disclosed, it will send a failure message.

- **TrustBuilder1-Relevant**: Disclose a credential C's policy only if C is syntactically relevant to R. A credential C is syntactically relevant to resource R iff C appears in R's policy, or C appears in the policy of a credential C' that is relevant to R.

Characterizing Safe Disclosure Sequences: An Example

- If Alice adopts **TrustBuilder1-Simple** strategy, then when she receives CPN's policy for exemption from sales tax, she will disclose all her unlocked credentials and her policies for her locked credentials (i.e., library card, and may ask CPN that in order to see her patient ID, CPN should present a certificate issued by UPMC).

- If CPN adopts the **TrustBuilder1-Relevant** strategy, CPN will identify that only her credit card and her reseller's license are relevant.



The Disclosure Tree Strategy Family

- Let M = (m_1, ….., m_k) be a sequence of messages such that $m_i \neq \emptyset$ and R doesn't belong m_i for $1 \leq i \leq k$. Let also L_A and L_B denote the local policies of parties Alice and Bob, respectively, and $S_M = U_{1 \leq i \leq k}\ m_i$.

- **DEFINITION 6.1.** The Disclosure Tree Strategy (DTS) is a strategy DTS(M, L_A, R) such that:

Condition 1 states under what circumstances the DTS strategy will terminate the negotiation with a failure message.

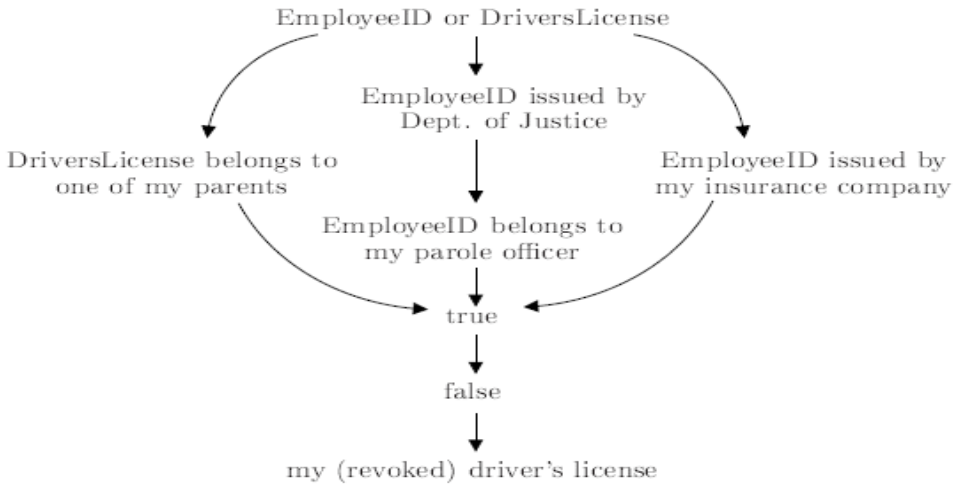
Condition 2 guarantees that the other party will have an evolvable tree.

THEOREM 6.1. ***The set of strategies generated by DTS is a family.***

  - m' is a non-empty set of credentials and policies such that view(S_M U m') contains at least one evolvable tree for Bob, and no non-empty proper subset of m' has this property.

Credentials with Internal Structure and Sensitive Policies

- Policies do in reality contain sensitive information.
- To protect sensitive polices, policy graphs can be used: *The essential idea is that the policy for a resource can be disclosed gradually*.
- Credentials can be modeled as a set of attribute-value pairs {$C.a_1$, = v_1, …., $C.a_k$ = v_k }, where a_i is an attribute name and v_i is an atomic uninterpreted value for that attribute, for $1 \leq i \leq k$.
- Credential names should not give away any information about credential contents.



Example Policy Graph

Credentials with Internal Structure and Sensitive Policies

DEFINITION 7.3. Suppose G is the policy graph of a resource R. Let P be a node in G and $P_0 \rightarrow \ldots \rightarrow P_k = P$ be a path from the source node P_0 to P. Such a path is called a ***policy path*** to P.

A policy path to P indicates when P can be safely disclosed. If a policy is sensitive, more ancestors can be added to P (thus more constraints) in G to control P's disclosure.

PROBLEM: When credentials have internal structure and no externally obvious key, when Alice receives Bob's disclosed policy P for a credential C, she can't tell what information P is protecting: the name "C" is now opaque. To which credential-valued variable in Alice's local polices C is to be bounded?

Binding!!

Alice will no longer have a clear picture of what Bob's disclosure trees would look like!!

Credentials with Internal Structure and Sensitive Policies

- The **DTS** family strategies can't interoperate anymore → **BTS** (Binding Tree Strategy).
- **TrustBuilder Protocol 1**→ **TrustBuilder Protocol 2** (the difference resides in the types of information contained in the messages).
- The family strategies they present require complete disclosure of the unlocked portion of policy graphs that are considered relevant to the negotiation.

Credentials with Internal Structure and Sensitive Policies

DEFINTION 9.1. A binding tree is a finite tree T satisfying the following conditions:

1) Each non-root node represents a credential, a service, a traversal of a policy graph, or else is a special node called a more-extensions node.

2) The root represents the resource ("service") to which access was originally requested.

3) Each credential/service node C has at most one child. The child, if present, must be a traversal node that represents a traversal of C's policy graph.

4) If a traversal node has children, they can either be credential nodes or a single more-extensions node.

5) If a traversal node's children are credential nodes, then they represent an extension to the traversal.

6) A more-extensions node has no children.

Operations: reduction, expansion, $\text{prune}_{\text{refusal}}$, $\text{prune}_{\text{redundant}}$, and view are all also defined.

Credentials with Internal Structure and Sensitive Policies

DEFINTION 9.8. Suppose Alice and Bob are the two negotiation parties. Let M = (m_1, …., m_k) be a sequence of messages such that $m_i \neq \emptyset$ and R doesn't belong m_i for $1 \leq i \leq k$. Let L_A and L_B be the local credentials, services and policy graphs of Alice and Bob respectively. Let $S_M = U_{1 \leq i \leq k} \; m_i$. Without loss of generality, assume it is Alice's turn to send the next message to Bob. The binding tree strategy (BTS) is a strategy such that BTS(M, L_A, R) satisfies the following conditions:

1) BTS(M, L_A, R) = {Ø} if and only if one of the following conditions holds:

   - View(S_M) = Ø.
   - There exists a set S' of safe disclosures that Alice can make such that view(S_M U S') = Ø.
   - Alice cannot expand S_M.

2) Otherwise, BTS(M, L_A, R) contains all the messages m' such that one of the following conditions holds:

   - m' = {R}, if Alice possesses R and R is unlocked by credentials disclosed in S_M.
   - m' is a minimal non-empty set of disclosures such that $S_M \cap m' = \emptyset$ and Bob can expand S_M U m'.

Strengths

1) They define a unique underlying protocol so that different strategies that belong to the same family can interoperate.
2) Safety conditions and timely termination of trust negotiations can be implicitly forced via disclosure trees (binding trees) no matter what policies and credentials the parties possess.
3) They easily formulate the concepts of negotiation protocols, strategies, and interoperation.
4) TrustBuilder Protocols (1 &2) are simple and natural.
5) A participant in negotiation can choose strategies based on its needs without worrying about interoperability (as long as it negotiates with other participant who uses strategies from the same family).

Weaknesses

1) The proposed protocols are static. What if a party adds a policy or remove one during negotiation.
2) The BTS family requires complete disclosure of the unlocked portion of policy graphs that are considered relevant to the negotiation. This lessens the degree of protection that policy graphs provide for sensitive policies.
3) Is modeling a credential as a set of attribute-value pairs the best option? (when Bob discloses C's policy P, they require him to disclose also what C is relevant to- to bind C- Ideally we want to decouple the disclosure and the satisfaction of a policy so that Alice can gain access to Bob's resource that is "very" sensitive without Alice's policies being ever disclosed).
4) Potential of denial-of-service attacks: Bob can't realize that Alice is attacking him (time-wasting trust negotiation).

Thank you!

Supporting Structured Credentials and S ensitive Policies through Interoperable St rategies for Automated Trust Negotiation

M. Hammoud

Dept. of Computer Science
University of Pittsburgh