Privacy-Preserving Trust Negotiations

Mikhail Atallah

Department of Computer Science
Purdue University

http://www.cerias.purdue.edu
Project info

- Collaborators (Ph.D. students):
  - Keith Frikken
  - Jiangtao Li

- Publications
  - NDSS ‘06
  - WPES ‘04
Access control

• Access control decisions are often based on requester characteristics rather than identity
  – Access policy stated in terms of attributes to be satisfied

• Digital credentials, e.g.,
  – Citizenship, age, physical condition (disabilities), employment (government, healthcare, FEMA, etc), credit status, group membership (AAA, AARP, ...), security clearance, ...
Scenario

- Requester ("Alice") sends request to resource owner ("Bob")
- Bob sends policy for access
- Alice sends appropriate credentials
- Bob verifies credentials and grant access if they satisfy policy
Problem: Sensitive info

- Treat credentials as sensitive
  - Better individual privacy
  - Better security

- Treat access policies as sensitive
  - Hide business strategy (fewer unwelcome imitators)
  - Less “gaming”
Previous work

• Mostly on sensitive credentials
• Minimizing credential disclosure
• Introduced “use-policies” for credentials
• Multiple rounds in a negotiation
  – A round makes more credentials usable
  – Stop when no change occurs in a round
Ideal Solution Requirements

• Neither side learns anything about the other’s credentials
• Neither side learns anything about the other’s policies
• Neither side learns which of their own credentials caused were relevant to successful negotiation
• No off-line probing (e.g., by requesting an M once and then trying various subsets of credentials)
Model

Server \( S = S_1, \ldots, S_m \) sends a request for \( M \), along with \( M, P_s \).

Protocol

\( C = C_1, \ldots, C_n \); \( P_c \)

Client

\( M \text{ if } C \) satisfies \( M \)’s policy

- \( M = \) message; \( P_c, P_s = \) Policies; \( C, S = \) credentials
  - Credential sets \( C \) and \( S \) are issued off-line, and can have their own “use policies” (included in \( P_s \) and \( P_c \))
- Client gets \( M \) iff her usable \( C_j \)’s satisfy access policy for \( M \)
- Cannot use a trusted third party
Credentials

- Generated by certificate authority (CA), using Identity Based Encryption
- E.g., issuing Alice a student credential:
  - Use Identity Based Encryption with ID = Alice||student
  - Credential = private key corresponding to ID
- Simple example of credential usage:
  - Send Alice M encrypted with public key for ID
  - Alice can decrypt only with a student credential
  - Server does not learn whether Alice is a student or not
Policy

- A Boolean function $p_M(x_1, \ldots, x_n)$
  - $x_i$ corresponds to attribute $attr_i$
- Policy is satisfied iff
  - $p_M(x_1, \ldots, x_n) = 1$ where $x_i$ is 1 iff there is a usable credential in C for attribute $attr_i$
- E.g.,
  - Alice is a senior citizen and has low income
  - Policy = $(\text{disability} \lor \text{senior-citizen}) \land \text{low-income}$
  - Policy = $(x_1 \lor x_2) \land x_3 = (0 \lor 1) \land 1 = 1$
Dependencies

- Credentials’ “use policies” can depend on each other
  - Arbitrarily deep nesting of dependencies
  - Cycles are possible
  - Cycles can overlap
- Trust negotiation process is iterative
Iterative negotiation

• Eager strategy
  – Policy-usability determination is done by a “forward flooding” process that determines which credentials are usable
  – Works for DAG only

• We use a “Reverse-Eager” Strategy
  – Can handle cycles
  – Use doubly-blinded policy evaluation
Iteration

• Phase 1: Credential and Attribute Hiding

• Phase 2: Blinded Policy Evaluation
Iteration overview

• For each $attr_i$, A generates 2 randoms $r_i[0], r_i[1]$

• Engages with B in sub-protocol the outcome of which is that B learns $n$ values $k_1, k_2, \ldots, k_n$

  $- k_i = r_i[1]$ if B has and can use a credential for $attr_i$, otherwise $k_i = r_i[0]$
Iteration overview (cont’d)

- Above hiding stage makes use of equality test for array elements
  - “Is there an index at which the array items are equal?”
  - Answer is split: A knows the randoms that encode “yes” and “no”, B learns one of those randoms

- Blinded policy evaluation
  - Uses the $k_1, k_2, \ldots, k_n$ and the $n$ pairs $r_i[0], r_i[1]$ computed in the hiding stage
  - Results in A’s usable credential set being *implicitly* updated according to whether policies evaluate to true
Other access control result

• Key-derivation
  – In access hierarchies [CCS 05, SACMAT 06]
  – New:
    • Time-based (discrete time units)
    • Geo-spacial based (planar regions)
    • Combinations of above

• In all of the above: 1 key, O(1) time derivation, almost-linear public storage
For more details ...

... on the trust negotiations result:
http://homes.cerias.purdue.edu/~jtli/paper/ndss06.pdf

or