Automated Trust Negotiation

William H. Winsborough, Kent E. Seamons
IBM Transarc Lab
Vicki E. Jones
North Carolina State University

Paper presented by Andrew K. Adams
Motivation


• “Winslett et al. recognized the potential to use server credentials to establish client trust ... they present detailed mechanisms for clients to submit credentials to servers ...”

• “... and note the relevance of such machinery for the reverse scenario, in which servers encourage clients by presenting their own credentials ...”
• “...Such a reversal provides a good basis for clients establishing the trust in servers required before disclosing sensitive credentials.”
Motivation, Cont.

• “... Such a reversal provides a good basis for clients establishing the trust in servers required before disclosing sensitive credentials.”

• “However, Winslett et al. is unclear about the details of how this kind of trust can be established.”
Terminology

- Security Agent (SA)
- Credential Access Policy (CAP)
  - defines the policy necessary to disclose a credential \( c \)
  - \( \text{gov}_{\text{client}}(c) \) or \( \text{gov}_{\text{server}}(c) \)
  - written in RAL (discussed later)
Terminology, Cont.

- Mobile Policy
  - an *automatic* distribution of policy
- Service-Governing Policy (SGP)
  - the policy (credential expression) that needs satisfied in-order to authorize service request
Terminology, Cont.

- Mobile Policy
  - an *automatic* distribution of policy
- Service-Governing Policy (SGP)
  - the policy (credential expression) that needs satisfied in-order to authorize service request

- Are there any scenarios where it’s bad if the server issues a mobile SGP?
TN Model

- Credential Expression ($\psi$)
  - “logical expression over credentials with constraints on their attributes”
  - in implementation, written in RAL
- Satisfied credential expression
  - if $\psi$ is satisfied by set of credentials, C
  - we write $\text{sat}(C, \psi)$
TN Model, Cont.

• Unlocked credentials
  • if $C \subseteq \text{ClientCreds}$ and $c \in \text{ServerCreds}$ such that $\text{sat}(C, \text{gov}_{\text{server}}(c))$ or
  • if $C \subseteq \text{ServerCreds}$ and $c \in \text{ClientCreds}$ such that $\text{sat}(C, \text{gov}_{\text{client}}(c))$
    • we write $\text{unlocked}(c, C)$
• Sequence of credential disclosures
  • $\{C_i\}_{i \in [0,2n+1]} = C_0, C_1, ... C_{2n+1}$
  • $C_{2i} \subseteq \text{ClientCreds}, C_{2i+1} \subseteq \text{ServerCreds}$
• Credential Request
  • credential expressions exchanged by participants

• Successful trust negotiation
  • if the client’s final disclosure satisfies $\Psi$, i.e., $\text{sat}(C_{2n}, \Psi)$
  • if the server’s final disclosure satisfies $\Psi$, i.e., $\text{sat}(C_{2n+1}, \Psi)$
TN Strategies

• Eager Strategy
  • Each participant sends *every* credential that is currently unlocked by the previously received credential request.
• Parsimonious Strategy
  
  • credential requests are exchanged to guide the negotiation toward satisfying a particular trust target $\Psi$ ...
  
  • ... when and if a request is sent that can be satisfied by unprotected credentials, those credentials are disclosed in the next stage ...
  
  • ... after a request is satisfied by a disclosure of unprotected credentials, the client then resends its prior request ... going through the request backwards.
• Hybrid
  • CAP specifies (via a flag) whether to use parsimonious or eager
  • begin with a eager phase to attempt to negotiate using only credentials flagged for eager strategy
  • if necessary, phase 2 starts parsimonious strategy (taking advantage of credentials disclosed in phase 1)
PAL

• Property-based Authentication Language
  • based on Trust Policy Language (TPL)
  • it defines one or more roles (a property of subjects defined in terms of the credentials they possess and the attribute values within)
  • issuer-role constraints and general constraints

\[
\text{role}(\text{Attributes}, \ldots) \leftarrow \text{CredentialVar}: \text{credentialType}, \ldots, \\
\text{role}_{1}(\text{credentialVar.issuer}, \text{Attributes}_{1}), \ldots, \\
\text{credentialConstraints}, \ldots
\]
PAL Example

hasCredit(Amount) ← LetterOfCredit: credit, creditor(LetterOfCredit.issuer), Amount = LetterOfCredit.amount.

creditor ← Creditor: creditor, self(Creditor.issuer).

shipper() ← Ref: reference, self(Ref.issuer), Ref.relationship = “shipper”.

shipper() ← Ref1: reference, Ref2: reference, Ref1 ≠ Ref2, knownClient(Ref1.issuer), knownClient(Ref2.issuer), Ref1.relationship = “shipper”, Ref2.relationship = “shipper”.
reputation(Rating) ← Membership: businessOrgMember,
    businessOrganization(Membership.issuer),
    Rating = Membership.rating.

newShippingClient(Pickup, Delivery) ← Destination: contract,
    Warehouse: lease
    knownBusiness(Destination.issuer),
    warehouseOwner(Warehouse.issuer),
    Pickup = Warehouse.location,
    Delivery = Destination.deliveryLocation.
• Role-based Authorization Language
  • a role-constraint expression, which expresses requirements for access to the service or credentials that it governs
  • these roles are defined by the authentication language PAL

\[
\text{hasCredit(Client, Amount) AND}
\text{Amount } \geq \text{Tons x costPerTon(PickupLocation, Destination)}
\]
clientWithAccount(Client, AccountNumber) OR
(reputation(Client, Rating)
  AND Rating ∈ \{good, excellent\} AND
  (knownClient(Client) OR
   (newShippingClient(Client, Pickup, Delivery) AND
    PickupLocation = Pickup AND Desitnation = Delivery)) AND
  hasCredit(Client, Amount) AND
  Amount ≥ Tons × costPerTon(PickupLocation, Destination))
CAP Example(s)

\[
\text{Contract}_C \equiv \text{shipper}(Server) \text{ AND} \\
\quad \text{reputation}(Server, \text{Rating}) \text{ AND} \ \text{Rating} \in \{ \text{good, excellent} \}
\]

\[
\text{Credit}_C \equiv \text{reputation}(Server, \text{Rating}) \text{ AND} \\
\quad \text{Rating} \in \{ \text{good, excellent} \}
\]

\[
\text{Warehouse}_C \equiv \text{shipper}(Server)
\]

\[
\text{B-Org-S}_C \equiv \text{true}
\]

\[
\text{Ref-1}_S \equiv \text{reputation}(Server, \text{Rating}) \text{ AND} \ \text{Rating} \in \{ \text{good, excellent} \}
\]

\[
\text{Ref-2}_S \equiv \text{reputation}(Server, \text{Rating}) \text{ AND} \ \text{Rating} \in \{ \text{good, excellent} \}
\]

\[
\text{B-Org-S}_S \equiv \text{true}
\]
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends B-Org-S to client
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends B-Org-S to client

Step 3. Client repeats service request and sends B-Org-C, Credit to server
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends B-Org-S to client

Step 3. Client repeats service request and sends B-Org-C, Credit to server

Step 4. Server sends B-Org-S, Ref₁ and Ref₂ to server
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends B-Org-S to client

Step 3. Client repeats service request and sends B-Org-C, Credit to server

Step 4. Server sends B-Org-S, Ref_1 and Ref_2

Step 5. Client repeats service request and sends B-Org-C, Credit, Contract and Warehouse
Putting it All Together

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends B-Org-S to client

Step 3. Client repeats service request and sends B-Org-C, Credit to server

Step 4. Server sends B-Org-S, Ref$_1$ and Ref$_2$

Step 5. Client repeats service request and sends B-Org-C, Credit, Contract and Warehouse

Step 6. Server authorizes request
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends SGP to client
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo.

Step 2. Server sends SGP to client.

Step 3. Client sends CAP for contract, credit & warehouse to server.
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends SGP to client

Step 3. Client sends CAP for contract, credit & warehouse to server

Step 4. Server sends CAP for rep$_1$ & rep$_2$
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 2. Server sends SGP to client

Step 3. Client sends CAP for contract, credit & warehouse to server

Step 4. Server sends CAP for rep₁ & rep₂

Step 5. Client repeats previous CAP and sends B-Org-C
All Together Now, Cont.

Step 1. Client sends request to schedule shipping 5 tons of cargo.

Step 3. Client sends CAP for contract, credit & warehouse to server.

Step 5. Client repeats previous CAP and sends B-Org-C.

Step 2. Server sends SGP to client.


Step 1. Client sends request to schedule shipping 5 tons of cargo

Step 3. Client sends CAP for contract, credit & warehouse to server

Step 5. Client repeats previous CAP and sends B-Org-C

Step 7. Client repeats initial request and sends B-Org-C, Credit, Contract and Warehouse

Step 2. Server sends SGP to client

Step 4. Server sends CAP for rep₁ & rep₂

Step 6. Server sends B-Org-S, Ref₁ and Ref₂
Step 1. Client sends request to schedule shipping 5 tons of cargo.

Step 3. Client sends CAP for contract, credit & warehouse to server.

Step 5. Client repeats previous CAP and sends B-Org-C.

Step 7. Client repeats initial request and sends B-Org-C, Credit, Contract and Warehouse.

Step 2. Server sends SGP to client.


Step 8. Server authorizes request.
Thoughts

• Assuming a usable PKI exists, would you trust your policies (CAPs & SGP, written in RAL/PAL) to correctly protect your sensitive credentials during automated trust negotiation sessions?
Conclusions

• I thought this paper was gutsy, i.e., suggesting we allow credentials (perhaps confidential) to be disclosed automatically.

• I didn’t like their handling of *Supporting Credentials*:
  • They assume that the submitter will hold all credentials.
  • Too much acceptance of self-signed ...