On interoperable trust negotiation strategies

S. Baselice, P.A. Bonatti, M. Faella

Giugno, 2007

\(^1\)Università di Napoli Federico II
Context

In Trust Negotiation Frameworks such as TRUST BUILDER, RT, PEER TRUST, PROTUNE

Transactions require

Access Control + Controlled Sensitive Disclosures

Trust Negotiations
Context

**Alice**

**Bob**

**Step 1:** Alice requests a service from Bob

**Step 2:** Bob discloses his policy for the service

**Step 3:** Alice discloses her policy for VISA

**Step 4:** Bob discloses his BBB credential

**Step 5:** Alice discloses her VISA card credential

**Step 6:** Bob grants access to the service

---

S. Baselice, P.A. Bonatti, M. Faella

On interoperable trust negotiation strategies
Context

**Alice**

- Step 1: Alice requests a service from Bob

- Step 2: Bob discloses his policy for the service

- Step 3: Alice discloses her policy for VISA

- Step 4: Bob discloses his BBB credential

- Step 5: Alice discloses her VISA card credential

- Step 6: Bob grants access to the service

**Bob**
Many Trust Negotiation Frameworks protect peers’ policies:

**Example**

- a bank grants special treatments to rich customers
- many other customers would not appreciate such privileges
A negotiation may fail

- because peers’ negotiation strategies don’t release all of the policy
- even if the peers’ policies permit a successful transaction
Our Goal

Guidelines for Negotiation Strategies that

1. make transactions succeed keeping partially secret both policies and sensitive information

Another goal:

2. reduce the amount of sensitive information released
Previous approaches:

- start from desirable "good" properties for negotiation strategies for designing a family of strategies that work well together.
Our Approach

Our approach:

- starts from the motivations that drive peers in releasing information for *deriving* negotiation strategies:
  - Servers want to publish services
  - Client want to access to services
  - *making transactions succeed*

As side effect we obtain a "good" property:

**Interoperability:** strategies yield a successful negotiation whenever the policies of the involved peers permit it.
Abstract Negotiation Framework

Policy language $\mathcal{L}$:
- a set of policy items
  - policy rules
  - portfolio: digital credentials, declarations
Abstract Negotiation Framework

**Policies + Portfolio** :
- finite subsets of $\mathcal{L}$
- all the information that a peer has for negotiating a resource

![Diagram of Negotiation Process]

**Steps**

1. Alice requests a service from Bob
2. Bob discloses his policy for the service
3. Alice discloses her policy for VISA
4. Bob discloses his BBB credential
5. Alice discloses her VISA card credential
6. Bob grants access to the service

S. Baselice, P.A. Bonatti, M. Faella

On interoperable trust negotiation strategies
Abstract Negotiation Framework

The semantics of policies is modelled by

\[ \text{unlocks} \subseteq \mathcal{P}(\mathcal{L}) \times \mathcal{L} \]

\( P \text{ unlocks } x \) iff \( P \) allows \( x \) to be released

**Monotonicity**: if we add more policy rules and credentials to a policy then the set of unlocked policy items increases [K. Seamons et al., *Requirements for policy languages for trust negotiation.*]

**Expressiveness**: 
\[ \forall q \in \mathcal{L} \text{ there exists a finite } P \subseteq \mathcal{L} \text{ s.t. } P \text{ unlocks } q \]
Abstract Negotiation Framework

**Messages:**
- a finite subset of $\mathcal{L}$
- information exchanged between a client and a server for negotiating a resource
- client’s requests for a resource
Abstract Negotiation Framework

*Peer*: a pair \( A = (P_A, R_A) \)

- \( P_A \): *policy + portfolio*
- \( R_A \): \( Msgs^* \rightarrow Msgs \) is a *release strategy*

Given the past history of negotiation, a *release strategy* prescribes the next "move" of a peer.
Abstract Negotiation Framework

Transaction $T = \langle A, B, \text{res}, F \rangle$

- $A$ (client) and $B$ (server) are peers;
- $\text{res} \in \mathcal{L}$ is a policy item (the initial request, $\text{res} \in P_B$);
- $F \subseteq \mathcal{M}gs^*$ is a failure criterion, i.e. the set of all possible failed negotiations.
Abstract Negotiation Framework

**Negotiation** $\text{nego}(T)$ induced by $T$, $R_A$ and $R_B$

- the finite or infinite sequence of messages $\mu = \mu_0\mu_1\ldots\mu_k\ldots$ mutually exchanged between $A$ and $B$
- $\mu_0 = \{\text{res}\}$
- $\text{nego}(T)$ terminates when
  - $\text{nego}(T) \in F$ (negotiation is *failed*)
  - res $\in \bigcup_{i=1}^{\left\vert \mu \right\vert} \mu_i$ (negotiation is *successful*)
Abstract Negotiation Framework

To get our results we have

- to restrict the class of peers that we study
- to fix a failure criterion

Negotiation Framework

\( \Psi = (C, F) \)

- \( C \) : a class of peers;
- \( F \) : a failure criterion.
Peers classification

**Truthful:** for all $hist$, $R_A(hist) \subseteq P_A$
- No item is "invented".

**Secure:** for all $hist$, $R_A(hist) \subseteq \text{unlocked}(P_A, hist)$
- The disclosure policy is preserved.

**Monotonic:** if $\text{released}(hist) \subseteq \text{released}(hist')$
- $R_A(hist) \subseteq R_A(hist')$
- The more information is received, the more information is released

**Monotonic servers are of practical interest**
- A better characterization of the client lets the server present a wider range of choices to get the desired resource.
Failure Criteria and Termination

Vacuous Messages
- equivalent to empty message;
- it carries no new information.

Failure criteria $F_k$
- a negotiation fails after $k$ consecutive vacuous messages.
Negotiation Framework

Next we focus on the negotiation framework

\[ \psi = (C, F_k) \]

\( F_k \): a failure criterion with \( k > 0 \)

\( C \):

- monotonic servers
- canonical (truthful and secure) peers
  - If \( A \) and \( B \) are truthful, termination is guaranteed.
Starting point: what do peers want?

Peers are **selfish**: their only goal is to make transactions succeed.

**Cooperativeness:**
- Cooperative peers are those whose strategies maximize the set of successful transactions.
Towards guidelines

_n-cautious peers_

- after $n$ vacuous messages
- if $A$ has something to release

\[
\text{unlocked}(P_A, \text{hist}) \not\subseteq \text{released}(\text{hist})
\]

- then $A$ releases something

\[
R_A(\text{hist}) \not\subseteq \text{released}(\text{hist})
\]

_weakly n-cautious peers_

- after $n$ vacuous messages
- if $A$ has something to release that could be useful
- then $A$ releases something.
Interacting with monotonic servers

Theorem

A peer $A$ is cooperative w.r.t. monotonic peers iff $A$ is $(k - 2)$-cautious.

- To make a client $A$ cooperative with monotonic servers, it is necessary and sufficient to program $A$’s strategy in a $(k - 2)$-cautious way.
- But how to make a monotonic server cooperative w.r.t. a $(k - 2)$-cautious client?
Interacting with $(k - 2)$-cautious peers

**Theorem**

A peer $B$ is cooperative with all $(k - 2)$-cautious peers iff $B$ is weakly $(k - 2)$-cautious.

To make a server $B$ cooperative with $(k - 2)$-cautious clients, it is necessary and sufficient to program $B$’s strategy in a weakly $(k - 2)$-cautious way.

**Note:** for efficiency it might be preferrable to adopt cautiousness as an approximation of weak cautiousness.
Summary

In any negotiation framework

- $\Psi = (C, F_k)$
- monotonic servers
- selfish peers (cooperative)

strategies must be

- $(k - 2)$-cautious on clients
- weakly $(k - 2)$-cautious on servers
Unexpected side effects

- each client is INTEROPERABLE with each server
- each client is INTEROPERABLE with each client

Interoperability:

- whenever a successful transaction is possible, the strategies find some
- even if the policies are partially kept secret
Further Guidelines

How to choose a value for parameter $k$ of $F_k$:

- $k$ even (to avoid exploits)
- preferrably $k = 2$

See the paper.
Future Work

Sensitivity Minimizing

- guidelines to program release strategies that minimize the amount of sensitivity of information disclosed during a negotiation
More on $k$ in $F_k$ - Even $k$ vs. Odd $k$

Odd values of $k$ allow exploits even if both $A$ and $B$ are $(k - 2)$-cautious

- $A$ may send vacuous messages until $B$ is forced to disclose something 2 steps before failure
- If $B$ sends a vacuous message 2 steps before failure, then it really means it can’t release anything else
- $A$ can still disclose something at the last step and keep the negotiation alive
- Very bad for privacy – deprecated
More on $k$ in $F_k$ - Even $k$ vs. Odd $k$

Even values are ok

- The peer that starts the vacuous sequence is also the peer that must release something 2 steps before failure
- Optimal value: $k = 2$
- No vacuous messages unless a peer really can’t release anything new
Negotiations

**Negotiation** $\text{nego}(T)$ induced by $T = \langle A, B, \text{res}, F_k \rangle$, $R_A$ and $R_B$

- the finite or infinite sequence of messages $\mu = \mu_0\mu_1...\mu_k...$
  s.t.
  - $\mu_0 = \{\text{res}\}$;
  - for all even $i \in \mathbb{N}$, $\mu_{i+1} = R_B(\mu_{\leq i})$;
  - for all odd $i \in \mathbb{N}$, $\mu_{i+1} = R_A(\mu_{\leq i})$;
  - for all $i \in \mathbb{N}$, if $\text{res} \in \mu_i$ or $\mu_{\leq i} \in F$, then $\mu = \mu_{\leq i}$.
Cooperativeness

A peer $A$ is *cooperative* w.r.t. a class of peers $\mathcal{C}$, if no $A'$ is s.t.

- $A$ and $A'$ have the same policy $P$,
- for all $B \in \mathcal{C}$ and all $\Psi$-transactions $T$ involving $A$ and $B$, $\text{val}(T) \leq \text{val}(T[A'/A])$,
- for some $B \in \mathcal{C}$ and some $\Psi$-transaction $T$ involving $A$ and $B$, $\text{val}(T) < \text{val}(T[A'/A])$. 

A peer $A$ is $n$-cautious if

- for all transactions $T$ involving $A$
- and all prefixes $\mu$ of $\text{nego}(T)$,
- if $\mu$ has a vacuous tail whose length is $\geq n$
- then

$$\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu) \Rightarrow R_A(\mu) \not\subseteq \text{released}(\mu)$$

(i.e., $R_A(\mu)$ is not vacuous)
weak \( n \)-cautiousness

A peer \( A \) is \textit{weakly} \( n \)-\textit{cautious} if

- for all transactions \( T \) involving \( A \)
- and all prefixes \( \mu \) of \( \text{nego}(T) \),
- if \( \mu \) has a vacuous tail whose length is \( \geq n \) and
- if \( R_a(\mu) \) is vacuous then \( T \) fails while
- \( T \) can be successful,
- then

\[
\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu) \Rightarrow R_A(\mu) \not\subseteq \text{released}(\mu)
\]

(i.e., \( R_A(\mu) \) is not vacuous)