

On interoperable trust negotiation strategies*

Sabrina Baselice Piero A. Bonatti Marco Faella
Università di Napoli
Via Cinthia, I-80126 Napoli, Italy
{baselice,bonatti,mfaella}@na.infn.it

Abstract

Among the many works on trust negotiation, only a few deal with negotiation strategies. These works are tailored to specific frameworks—so their results cannot be extended to competing approaches—and introduce assumptions that cannot be always guaranteed. In this paper we identify some guidelines for designing “good” (interoperable) trust negotiation strategies under a different set of assumptions, namely, a peer’s interest in making transactions succeed. Moreover, since our analysis is based on an abstract framework, the guidelines apply to a wide range of policy languages and negotiation frameworks.

1 Introduction

Trust negotiation [8, 7, 11, 10, 9, 3, 6, 5, 1, 4] is an approach to security and privacy preserving interactions in open networked environment such as the web. In such scenarios peers often interact without any previous relationship and without sharing any common security domain. As a consequence, traditional authentication is sometimes undesirable and frequently impossible. Access control policies and privacy policies are rather based on user *properties*, possibly including user identity as a special case. Such properties can be encoded in various ways, including digital credentials, unsigned declarations, and reputation measures [2].

Negotiations arise because peers incrementally exchange such pieces of information to achieve a mutual level of trust that suffices to complete a transaction. To communicate security and privacy requirements, peers exchange also suitable subsets of their policies. Parts of the policies are not disclosed because they are irrelevant to the current transaction. Other fragments of the policies may be protected because the policies themselves may be sensitive [3, 4, 13]. In

some approaches, policy filtering may be a quite articulated process [4].

A crucial problem in automating trust negotiations is the choice of appropriate negotiation strategies. Some of the major related issues concern:

- *Interoperability*: Protecting policies means that the requirements to be fulfilled are not entirely published. Consequently, a peer may fail to disclose the right credentials because it does not realize that those credentials would grant the desired resource. Then the question is: to what extent policy hiding may affect negotiation success? Which strategies yield a successful negotiation whenever the policies of the involved peers permit it?
- *Confidentiality*: Is it possible to minimize the amount and the sensitivity of the information released during a negotiation?

While all works on TN specify the space of possible choices at each negotiation step, only a few works deal with strategies.

In [12, 13], strategies and interoperability are studied in the framework of the Trustbuilder protocols. A maximal family of mutually interoperable strategies—called DTS family—is introduced, which is closed under so-called *hybrid strategies* and includes two strategies of practical interest. The nice properties of the DTS family follow from two important assumptions: first, all the policy information that can be released at a given negotiation step is immediately released; second, when a peer does not want to disclose a credential, the peer declares it explicitly. These assumptions give peers a relatively accurate view of the available options for completing the negotiation. Unfortunately, in some scenarios there may be no guarantee that the peers actually meet these two assumptions.

In this work we aim at a general analysis of interoperability issues, independent from any particular policy language, in order to make our results applicable to a wide range of frameworks. Moreover, we are interested in strategies that do not necessarily meet the assumptions underlying the DTS family. In our framework, peers are motivated

*Partially supported by the network of excellence REVERSE, IST-2004-506779.

to release information only by their interest in making transactions succeed (roughly speaking, servers want to publish and clients want to access). We formalize such motivation through a notion of *cooperativeness* based on the game-theoretic notion of dominance. One of our main goals is understanding the minimal requirements on strategies that follow from cooperativeness. In other words, rather than *designing* a family of strategies that work well together (as in [12, 13]), we *derive* strategies from the motivations that drive peers.

Unfortunately, almost no positive result can be proved in a completely general setting, because unrestricted peers can behave in the strangest possible ways. For this reason—after some general negative results—we shall focus on a particular class of scenarios of practical interest, where servers adopt *monotonic strategies*. Such peers generalize typical commercial servers such that:

- their credentials can be freely published, because they certify properties that may attract customers, such as seal programs membership (Better Business Bureau, ETrust, etc.) and quality certifications;
- the more information is released by a client, the larger is the portion of policy disclosed by the server. Intuitively, a better characterization of the client lets the server present a wider range of choices to get the desired resource.

Accordingly, monotonic strategies disclose more information to peers that release more information about themselves.

On the contrary, clients—that are often requested to disclose sensitive user information—tend to adopt non-monotonic strategies. Knowing more about the server’s policy means having a wider range of opportunities for fulfilling that policy. In general, a wider range of choices lets the client satisfy the policy by releasing less sensitive credential sets. So, given more knowledge about the server’s policy, a client may decide to release a completely different set of information items.

Still, a client may adopt a (nonmonotonic) strategy which is *cooperative* with respect to monotonic strategies, in order to increase the chances of successful transactions with monotonic peers.

Under this assumption, we can derive the properties of the strategies that clients and monotonic servers should adopt in order to be cooperative (and possibly interoperable). Such properties provide guidelines to implement concrete trust negotiation strategies in a variety of frameworks.

The paper is organized as follows: In the next section we define the abstract negotiation framework. In Section 3 we prove our results. Section 3.3 deals with transactions that fail as soon as one peer sends a *vacuous* message, that is, a message that carries no new information. Section 3.4

deals with transactions that fail after $k > 1$ consecutive vacuous messages. To strengthen our results, we then consider monotonic servers. Section 3.5 characterizes the clients that are cooperative with monotonic servers, and Section 3.6 characterizes servers that are cooperative to such clients. A comparison of some termination criteria (Section 3.7) and some conclusions (Section 4) complete the paper.

Due to space limitations only some proofs are included.

2 An abstract negotiation framework

2.1 The policy language

Our negotiation framework abstracts away the details of the policy specification language and its semantics. In this way, our results apply to many such languages.

We assume as language \mathcal{L} any set, whose members will be called *policy items*. Intuitively, policy items comprise both policy rules (e.g., logic programming rules such as those adopted in [3, 5, 1, 4]), digital credentials, declarations [3, 4], and any resource and information that can be exchanged during negotiations.

By releasing a set of policy items m_1 to a peer A , a peer B may let A disclose a set of items m_2 . In this case we say that m_1 and the policy of A together *unlock* m_2 . Accordingly, the semantics of \mathcal{L} is modelled by a relation

$$\text{unlocks} \subseteq \wp(\mathcal{L}) \times \mathcal{L}$$

with the following properties:

Monotonicity: if $P_1 \subseteq P_2$ then P_1 unlocks p implies P_2 unlocks p .

Expressiveness: for all $q \in \mathcal{L}$ there exists a finite $P \subseteq \mathcal{L}$ such that P unlocks q .

The former intuitively says that as more information becomes available the mutual level of trust increases, and hence more items can be released (although a negotiation strategy may decide not to release them all). This property derives from the monotonicity requirement discussed in [7]. The second property means that the language is rich enough to define conditions for releasing any possible item (be it a rule, a credential, or a declaration). It is satisfied by many concrete frameworks, including [13, 5, 4].

Relation *unlocks* does not only model access control decisions. In some frameworks, a credential can be released only if the access to a requested resource explicitly depends on the credential (i.e., the credential should be *relevant* to the current negotiation). Relevance can be modelled via *unlocks*, too. In that case, R unlocks p intuitively means that (some of) the items in R (the information released during negotiation) satisfy the access control policy for item

p and p is relevant to the current negotiation (i.e., the elements of R show a connection between p and a requested resource).

Definition 2.1 A *policy* is a finite subset of \mathcal{L} . ■

What we call “policy” for simplicity actually covers also the portfolio of credentials and declarations [10, 3, 5, 4]. So, intuitively, a policy is all the information that a server (or a client) has for negotiating a resource. In some concrete frameworks, \mathcal{L} is a logic programming language, and policies are equivalent to Datalog programs.

2.2 Negotiations and strategies

Definition 2.2 A *message* is a finite subset of \mathcal{L} . ■

By definition 2.2, both messages and policies have the same structure. Messages represent information that is exchanged between a client and a server negotiating a resource. The client’s request for a resource is itself a message.

We shall denote by Msgs the set of all messages. Given a (possibly infinite) message sequence $\mu = \mu_0\mu_1 \dots \mu_i \dots$ we denote by $|\mu|$ the *length* of μ ($|\mu| \in \mathbb{N} \cup \{\infty\}$). Each prefix $\mu' = \mu_0 \dots \mu_j$ of μ will be denoted by $\mu_{\leq j}$ or $\mu_{< j+1}$. We say that μ *extends* μ' if μ' is a prefix of μ .

Definition 2.3 A *release strategy* is a function $R : \text{Msgs}^* \rightarrow \text{Msgs}$. ■

Given a sequence of messages, a release strategy prescribes the next “move” of a *peer*.

Definition 2.4 A *peer* (or *agent*) is a pair $A = (P_A, R_A)$, where P_A is a policy and R_A is a release strategy. ■

Definition 2.5 A *transaction* is a structure $T = \langle A, B, \text{res}, F \rangle$ where:

- A and B are peers, called the *client* and the *server* of T , respectively; A and B are also called the peers *involved* in T ;
- $\text{res} \in \mathcal{L}$ models the *initial request* that A submits to B ; res must belong to P_B ;
- $F \subseteq \text{Msgs}^*$ is a *failure criterion*; informally, it contains all the sequences of messages corresponding to a failed negotiation. ■

The fact that res belongs to P_B rules out some trivial transactions and makes it possible to simplify notation and theorem statements. The rationale behind it is that servers typically publish (possibly through a broker) the list of services and resources they deploy. Clients ask only for such services and resources. The services and resources published by a server are not completely unreachable, that is, they can be obtained by fulfilling suitable conditions.

Each transaction $T = \langle A, B, \text{res}, F \rangle$ induces a *negotiation* $\text{nego}(T) \in \text{Msgs}^* \cup \text{Msgs}^\omega$, that is a message exchange between the peers A and B .

Definition 2.6 Let $T = \langle A, B, \text{res}, F \rangle$ be a transaction. The *negotiation* induced by T , $\text{nego}(T)$, is the finite or infinite sequence of messages $\mu = \mu_0\mu_1 \dots \mu_k \dots$ that satisfies the following conditions:

- $\mu_0 = \{\text{res}\}$; this message is the initial request that A sends to B ;
- for all even $i < |\mu|$, $\mu_{i+1} = R_B(\mu_{\leq i})$;
- for all odd $i < |\mu|$, $\mu_{i+1} = R_A(\mu_{\leq i})$;
- for all $i \leq |\mu|$ such that $i \in \mathbb{N}$, if $\text{res} \in \mu_i$ or $\mu_{\leq i} \in F$, then $\mu = \mu_{\leq i}$ (i.e., the negotiation is either infinite, or it terminates as soon as some termination condition is met). ■

For all message sequences μ (including negotiations), $\text{released}(\mu) = \bigcup_{i=1}^{|\mu|} \mu_i$ shall denote the set of all policy items released by μ (note that the request message μ_0 is skipped). For all policies P , let

$$\text{released}(P, \mu) = P \cap \text{released}(\mu).$$

Analogously, let $\text{unlocked}(P, \mu)$ denote the items of P unlocked by the items released by μ :

$$\text{unlocked}(P, \mu) = \{p \mid p \in P \text{ and } (P \cup \text{released}(\mu)) \text{ unlocks } p\}.$$

A negotiation succeeds if the initial request is eventually released by the server. Formally:

Definition 2.7 A negotiation $\text{nego}(T)$ is *successful* if $\text{res} \in \text{released}(\text{nego}(T))$ and *failed* otherwise. A transaction T is *successful* if $\text{nego}(T)$ is, and *failed* otherwise. ■

2.3 Classifying and comparing peers

Peers can be classified according to the properties of their release strategies. A peer A is:

Truthful: if for all $\mu \in \text{Msgs}^*$, $R_A(\mu) \subseteq P_A$. Informally, truthful peers release only items they actually have. If an item represents a credential, then this assumption can be enforced via cryptographic techniques (it simply means that credentials are unforgeable). A similar consideration applies when the item represents the policy rule of a third party and the rule is signed (cf. [5]). If an item represents a local policy rule, then there may be no guarantee that the item is actually part of A ’s policy. Peer A might publish the item just to encourage other peers to release their information to satisfy the fake policy. However, it is easy to detect that a rule is not applied after its preconditions have been satisfied; reputation-based mechanisms (coupled with cautious trust-based protection of sensitive items) may reduce and confine the effectiveness of these attacks.

Secure: if for all $\mu \in \text{Msgs}^*$, $R_A(\mu) \subseteq \text{unlocked}(P_A, \mu)$.
That is, only unlocked information is released.

Monotonic: if for all message sequences μ_1 and μ_2 , $\text{released}(\mu_1) \subseteq \text{released}(\mu_2)$ implies $R_A(\mu_1) \subseteq R_A(\mu_2) \cup \text{released}(\mu_2)$. This means that the more information is released during a negotiation, the larger the set of items released by A at the next step. The term $\text{released}(\mu_2)$ is needed because in general, for efficiency, policy items may be sent only once during a negotiation and not repeated again and again in each message.

A peer is *canonical* if it is both truthful and secure.

We adopt the following notion of framework to focus on particular classes of peers and failure criteria.

Definition 2.8 A *negotiation framework* is a pair $\Psi = (\mathcal{C}, F)$ where \mathcal{C} is a class of peers and F is a failure criterion. A Ψ -peer is a member of \mathcal{C} . A Ψ -transaction is any $T = \langle A, B, \text{res}, F' \rangle$ such that A and B belong to \mathcal{C} and $F' = F$. The framework is *canonical* if \mathcal{C} is a class of canonical peers. ■

Next, we formalize the assumption that, other things being equal, peers are interested in achieving a successful negotiation. To this purpose, we employ the game-theoretic notion of *domination*.

Let $\text{val}(T) = 1$ if T is successful and $\text{val}(T) = 0$ otherwise. For all $T = \langle A, B, \text{res}, F \rangle$ let $T[A'/A] = \langle A', B, \text{res}, F \rangle$ and $T[B'/B] = \langle A, B', \text{res}, F \rangle$.

Definition 2.9 A peer A Ψ -*dominates* a peer A' w.r.t. a class of peers \mathcal{C} if

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

Informally, A dominates A' if they have the same policy and the strategy of A is strictly better than the strategy of A' (in terms of the successful transactions it yields). Now we can define cooperative peers as those whose strategy maximizes the set of successful transactions:

Definition 2.10 A peer A is Ψ -*cooperative* w.r.t. a class of peers \mathcal{C} , if no A' Ψ -dominates A in \mathcal{C} . When \mathcal{C} comprises all Ψ -peers, we simply say that A is Ψ -cooperative. ■

The notion of cooperativeness does not enjoy many closure properties. Suppose A is cooperative w.r.t. a class \mathcal{C} . It is not hard to see that if $\mathcal{C} \subseteq \mathcal{C}'$, then A is not necessarily cooperative w.r.t. \mathcal{C}' (roughly speaking, A might not interact optimally with some $B \in \mathcal{C}' \setminus \mathcal{C}$). Similarly, if $\mathcal{C} \supseteq \mathcal{C}'$, then A is not necessarily cooperative w.r.t. \mathcal{C}' (some A' might dominate A over \mathcal{C}' , and give worse results interacting with some $B \in \mathcal{C} \setminus \mathcal{C}'$). We only have:

Proposition 2.11 If A is cooperative w.r.t. \mathcal{C}_1 and \mathcal{C}_2 , then A is cooperative w.r.t. $\mathcal{C}_1 \cup \mathcal{C}_2$.

The proposition is not valid if we replace \cup with \cap .

2.4 Interoperability

Informally, two peers are interoperable if they can successfully interact, provided that *their policies allow it*.

The message sequences defined below capture what the policies allow to disclose: Each message contains only information unlocked by the previous messages, so the policies are satisfied. Note that in general a strategy may be more restrictive. For example, it may release an item only if that item has been explicitly requested (no blind disclosures to meet hidden policies).

Definition 2.12 Let P_A, P_B be policies and res be a request. A *disclosure sequence* for P_A, P_B and res is a $\mu \in \text{Msgs}^*$ such that

- $\mu_0 = \{\text{res}\}$,
- for all even i such that $0 < i \leq |\mu|$, $\mu_i \subseteq \text{unlocked}(P_A, \mu_{<i})$,
- for all odd i such that $i \leq |\mu|$, $\mu_i \subseteq \text{unlocked}(P_B, \mu_{<i})$.

A *successful disclosure sequence* μ for P_A, P_B and res is a disclosure sequence s.t. $\text{res} \in \text{released}(\mu)$. A successful disclosure sequence for $T = \langle A, B, \text{res}, F \rangle$ is a successful disclosure sequence for P_A, P_B and res . ■

We are now ready to formulate the notion of interoperability in terms of our framework.

Definition 2.13 Two peers A and B are Ψ -*interoperable* if for all Ψ -transactions T involving A and B , T has a successful disclosure sequence only if T is successful. ■

We shall sometimes omit Ψ when it is clear from the context.

Remark 2.14 The notion of disclosure sequence and a similar formalization of interoperability have been introduced in [12, 13]. However, in [12, 13], the notion of interoperability is not always symmetric. It becomes symmetric in strategy *families*.

In canonical frameworks, interoperability is stronger than cooperativeness.

Theorem 2.15 Let Φ be any canonical framework. For all classes of peers \mathcal{C} , if A is Φ -interoperable with all peers in \mathcal{C} , then A is Φ -cooperative w.r.t. \mathcal{C} .

3 Properties of negotiation frameworks

The following terminology will be useful in the proof of our results. Given a transaction T with client A and server B , for all prefixes $\mu_{\leq i}$ of $\text{nego}(T)$, let $\text{turn}_T(\mu_{\leq i}) = B$ if i is even, and $\text{turn}_T(\mu_{\leq i}) = A$ if i is odd. We say that μ' is an A -prefix of μ if μ' is a prefix of μ and $\text{turn}_T(\mu') = A$. We say that T visits a sequence of messages μ if μ is a prefix of $\text{nego}(T)$.

3.1 Failure criteria and termination

Consider the following family of failure criteria. For all $k > 0$, let

$$F_k = \{\mu = \mu_0 \dots \mu_n \mid \mu_{n-k+1} \cup \dots \cup \mu_{n-k+n} \subseteq \text{released}(\mu)\}.$$

Intuitively, F_k stipulates that a negotiation fails as soon as the peers exchange k messages that bear no new information, that is, k consecutive *vacuous messages*:

Definition 3.1 Let $\mu = \mu_0 \mu_1 \dots \mu_k$ be a negotiation w.r.t. a transaction T . We say that a message m is *vacuous* w.r.t. μ (or simply *vacuous* if there is no ambiguity) if $m \subseteq \text{released}(\mu)$.

The *vacuous tail* of μ is the longest suffix of μ consisting only of vacuous messages. ■

In [12, 13] the termination criterion is F_1 . The same authors consider F_k with $k > 1$ in an unpublished manuscript. A nice property of this family of failure criteria is that negotiations terminate under a mild assumption.

Lemma 3.2 For all transactions $T = \langle A, B, \text{res}, F \rangle$, if A and B are truthful and $F = F_k$, then $\text{nego}(T)$ is finite.

We shall focus on the negotiation frameworks $\Psi_k = (\mathbb{C}, F_k)$, where \mathbb{C} is the set of all canonical peers. Note that if $k = 0$, then all sequences are failed.

3.2 An operational criterion

We are going to prove that cooperativeness implies some operational constraints on strategies, that we formalize by the notion of *cautiousness* below. In particular, cautiousness means that as failure is getting closer, a peer must release some of its unlocked information (if any). Such operational constraints constitute guidelines for designing concrete strategies. We shall prove that in a large class of practically interesting scenarios, satisfying the operational constraints is a necessary and sufficient condition for a strategy to be cooperative, as well as interoperable.

Definition 3.3 Let $n \in \mathbb{N}$ and $\nu \in \text{Msgs}^*$. A peer A is *n-cautious after ν* if for all transactions T involving A and all sequences μ such that:

- i. μ is an A -prefix of $\text{nego}(T)$ extending ν ,
- ii. μ has a vacuous tail whose length is n or greater,

it holds that $\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu)$ implies $R_A(\mu) \not\subseteq \text{released}(\mu)$ (i.e., $R_A(\mu)$ is not vacuous if at least one unlocked unreleased item exists).

We say that A is *n-cautious* if it is n -cautious after the empty sequence. ■

In other words, after n vacuous messages, an n -cautious peer is forced to send a non-vacuous message (if possible).

3.3 Frameworks with termination criterion F_1

In this section, we consider the negotiation framework Ψ_1 . Recall that Ψ_1 prescribes that a negotiation is failed as soon as a vacuous message is sent by either peer.

The first result states that peers that are Ψ_1 -cooperative do not send vacuous messages unless they have no other choice.

Theorem 3.4 If A is Ψ_1 -cooperative, then A is 0-cautious in every Ψ_1 -transaction where A is the client.

The same conclusion holds if A is the server. In that case, A is dominated by the server A' that, after the prefix μ , starts to release all unlocked information. We thus have the following.

Theorem 3.5 If A is Ψ_1 -cooperative, then A is 0-cautious in every Ψ_1 -transaction where A is the server.

Next, we show that two peers that are both Ψ_1 -cooperative need not be interoperable.

Example 3.6 Consider peers A and B , where $P_A = \{p, q\}$ and $P_B = \{\text{res}\}$. Suppose that \emptyset unlocks p , \emptyset unlocks q , and q is necessary and sufficient to unlock res (i.e., X unlocks res if and only if $q \in X$). The release strategy R_A is such that A releases first p and then q . The release strategy R_B simply releases res as soon as it is unlocked. Such peers are canonical.

For the transaction $T = \langle A, B, \text{res}, F_1 \rangle$, we have that T has a successful disclosure sequence (such as $\{\text{res}\} \emptyset \{q\} \{\text{res}\}$), but T is failed.

Peer B is obviously cooperative. To see that also A is cooperative, compare it with the peer A' that has the same policy as A , but whose release strategy chooses to release both p and q as soon as possible (at step 2). It turns out that A' does not dominate A , since there is a server B' that prefers A' over A : it is the server that releases res at step 5 (and releases irrelevant items in the meanwhile).

Finally, we show that peers that are Ψ_1 -cooperative w.r.t. monotonic peers always release all unlocked information. Clearly, this is an undesirable property that renders release strategies useless.

Theorem 3.7 *If A is Ψ_1 -cooperative w.r.t. monotonic peers, then for all Ψ_1 -transactions T where A is the client and the server is monotonic, and for all A -prefixes μ of $\text{nego}(T)$, we have $\text{unlocked}(P_A, \mu) \subseteq \text{released}(\mu) \cup R_A(\mu)$.*

3.4 Frameworks with termination criterion F_k , $k > 1$

In this section, when not explicitly stated otherwise, we consider an arbitrary negotiation framework $\Phi = \Psi_k$, with $k > 1$.

In general, if peers have no special property besides cooperativeness, then we can only prove a limited set of results. We start with a useful guarantee. Suppose the server is cooperative; if the client succeeds in unlocking the resource, then the server will release it (although this might not happen immediately).

Lemma 3.8 *Let \mathcal{C} be any class of peers. For all Φ -transactions $T = \langle A, B, \text{res}, F_k \rangle$ such that (i) A belongs to \mathcal{C} , (ii) B is cooperative w.r.t. \mathcal{C} , and (iii) $\text{res} \in \text{unlocked}(P_B, \text{nego}(T))$, we have that T is successful.*

From an operational point of view (cautiousness), cooperativeness requires that peers make a “last minute” attempt at preventing failure, that is, they must release some unlocked item (if any) if the negotiation is just one step away from failure. This property is formalized by the next two lemmas.

Theorem 3.9 *If A is Φ -cooperative, then A is $(k - 1)$ -cautious in every Φ -transaction.*

Unfortunately, the property of being $(k - 1)$ -cautious in itself is not very strong. It does not ensure cooperativeness nor interoperability. The class of $(k - 2)$ -cautious peers enjoys a number of nice properties, instead.

We start their analysis with a technical lemma, stating that if two peers are $(k - 2)$ -cautious from some step on, then the two peers are interoperable in a restricted sense, that is, for the particular initial request res that originates the negotiation.

In the following, when the context clearly states that μ is a prefix of $\text{nego}(T)$, where $T = \langle A, B, \text{res}, F \rangle$, we adopt the abbreviation:

$$\text{unlocked}(\mu) = \text{unlocked}(P_A, \mu) \cup \text{unlocked}(P_B, \mu).$$

Lemma 3.10 *For all Φ -transactions $T = \langle A, B, \text{res}, F_k \rangle$ and all prefixes ν of $\text{nego}(T)$ such that the last message of ν is not vacuous, if A and B are $(k - 2)$ -cautious after ν and T has a successful disclosure sequence, then T is successful.*

Proof: Suppose not, that is, for some $T = \langle A, B, \text{res}, F_k \rangle$ and some prefix ν of $\text{nego}(T)$, such that A and B are $(k - 2)$ -cautious after ν and T has a successful disclosure sequence μ , T is not successful. We shall derive a contradiction.

Let $\nu' = \text{nego}(T)$. By Lemma 3.2, ν' is finite. Suppose its length is n . Since $k > 1$, ν'_{n-1} and ν'_n must be vacuous, and hence $\text{released}(\nu'_{<n-1}) = \text{released}(\nu')$.

Clearly, the successful sequence μ releases some item that does not belong to $\text{released}(\nu'_{<n-1})$. So, let i be the least integer such that there exists $p \in \mu_i \setminus \text{released}(\nu'_{<n-1})$. Since i is minimal, $\text{released}(\mu_{<i}) \subseteq \text{released}(\nu'_{<n-1})$. Moreover, by definition of disclosure sequence, p must belong to $\text{unlocked}(\mu_{<i})$. Then, by monotonicity of unlocks, $p \in \text{unlocked}(\nu'_{<n-1})$. But then p should belong to either ν'_{n-1} or ν'_n , because A and B are $(k - 2)$ -cautious at that point. Consequently, p should be a member of $\text{released}(\nu')$ and hence of $\text{released}(\nu'_{<n-1})$: a contradiction. ■

As an immediate consequence, we have that $(k - 2)$ -cautious peers can interoperate with each other:

Theorem 3.11 *If A and B are $(k - 2)$ -cautious Φ -peers, then A and B are interoperable.*

While cooperative peers are not necessarily $(k - 2)$ -cautious, in general, we shall see in the next section that $(k - 2)$ -cautiousness is essential in order to interact with monotonic servers.

3.5 Interacting with monotonic servers: $(k - 2)$ -cautiousness

In this section we still assume that the underlying negotiation framework is some $\Phi = \Psi_k$ with $k > 1$.

We start this section with a technical lemma. Intuitively, it proves that monotonic peers “appreciate” that the other peers release some information, to the extent that within two steps from failure, a vacuous message may cause failure while a non-vacuous message may lead to success.

Lemma 3.12 *Let T be a Φ -transaction involving peers A and B . Suppose $\text{nego}(T)$ has a prefix μ with a $(k - 2)$ steps long vacuous tail and such that $\text{turn}_T(\mu) = A$. Then there exists a monotonic Φ -peer B' such that*

1. *If $R_A(\mu)$ is vacuous, then $T[B'/B]$ fails;*
2. *If $R_A(\mu)$ is not vacuous and $R_A(\mu \cdot \sigma) = \text{unlocked}(P_A, \mu \cdot \sigma)$, for all $\sigma \in \text{Msgs}^2$, then $T[B'/B]$ succeeds.*

Now we can derive an important operational guideline: In order to be cooperative with monotonic peers, it is necessary and sufficient that the strategy is $(k - 2)$ -cautious. We start by proving necessity.

Theorem 3.13 *If A is Φ -cooperative w.r.t. monotonic peers, then A is $(k - 2)$ -cautious.*

We are only left to prove that $(k - 2)$ -cautiousness suffices to be cooperative with monotonic peers.

Theorem 3.14 *If a peer A is $(k - 2)$ -cautious then A is Φ -cooperative w.r.t. monotonic peers.*

Proof: Suppose not. Then there exist a $(k - 2)$ -cautious peer A , a monotonic Φ -peer B , and a Φ -peer A' such that some transaction T involving A and B fails, but $T[A'/A]$ is successful. Let $\sigma = \text{nego}(T[A'/A])$ be the successful negotiation and $\phi = \text{nego}(T)$ be the failed one. Clearly, $\text{released}(\sigma) \setminus \text{released}(\phi) \neq \emptyset$ (it contains at least res), and hence there must be a positive index i such that $\sigma_i \not\subseteq \text{released}(\phi)$.

Let j be the least positive index such that $\sigma_j \not\subseteq \text{released}(\phi)$, and let p be any item such that $p \in \sigma_j \setminus \text{released}(\phi)$. We have $p \in \text{unlocked}(\sigma_{<j})$, because both A' and B are Φ -peers (and hence secure). This implies $p \in \text{unlocked}(\phi)$, because unlocks is monotonic and $\text{released}(\sigma_{<j}) \subseteq \text{released}(\phi)$ (by the minimality of j).

Now, if $p \in P_A$, then the last message of A in ϕ should not have been empty, because there would be at least one unlocked and unreleased item in A 's policy (i.e., p) and A is $(k - 2)$ -cautious. Then $p \in P_B$ must hold. Infact, since B is monotonic and $\text{released}(\sigma_{<j}) \subseteq \text{released}(\phi) = \text{released}(\phi_{\leq|\phi|-2})$ then we have that $R_B(\sigma_{<j}) \subseteq (R_B(\phi_{\leq|\phi|-2}) \cup \text{released}(\phi_{\leq|\phi|-2}))$. So, it should hold that $p \in (R_B(\phi_{\leq|\phi|-2}) \cup \text{released}(\phi_{\leq|\phi|-2}))$ and then $p \in \text{released}(\phi)$. A contradiction. ■

Unfortunately, being $(k - 2)$ -cautious is not sufficient to be *interoperable* with all monotonic peers, as the following simple example shows.

Example 3.15 Consider peer $B = (P_B, R_B)$, such that $P_B = \{\text{res}\}$, and $R_B(\mu) = \emptyset$ for all $\mu \in \text{Msgs}^*$. Notice that B is canonical and monotonic. Assume that \emptyset unlocks res . For all peers A and all $k > 0$, the transaction $T = \langle A, B, \text{res}, F_k \rangle$ admits the successful disclosure sequence $\{\text{res}\}\{\text{res}\}$. However, T is failed due to the restrictive release strategy of B . Thus, A and B are not interoperable.

For this reason we are going to put some restrictions on servers and investigate what happens when they are cooperative w.r.t. clients. But first we state a nice side effect of our approach: If two clients are cooperative with monotonic servers, then they are also interoperable with each other.

Corollary 3.16 *If A and B are Φ -cooperative w.r.t. monotonic peers, then A and B are Φ -interoperable.*

3.6 Weak cautiousness

Next we focus on the strategies that should run on a monotonic server, assuming we only require the server to be cooperative with respect to the clients that are in turn cooperative with monotonic servers (and hence $(k - 2)$ -cautious). The first result tells that in this context, every cooperative strategy guarantees interoperability.

Theorem 3.17 *If A is Φ -cooperative w.r.t. $(k - 2)$ -cautious peers, then A is Φ -interoperable with all $(k - 2)$ -cautious peers.*

Proof: Suppose not. Then there exists a transaction $T = \langle A, B, \text{res}, F_k \rangle$ (or $T = \langle B, A, \text{res}, F_k \rangle$) involving A and some $(k - 2)$ -cautious peer B s.t. there is a successful disclosure sequence for A, B and res but $\text{val}(T) = 0$.

Let $A' = (P_A, R_{A'})$ be the peer defined by

$$R_{A'}(\sigma) = \text{unlocked}(P_A, \sigma).$$

By construction, A' is $(k - 2)$ -cautious. Our aim is to derive the contradiction that A' dominates A w.r.t. $(k - 2)$ -cautious peers.

By Theorem 3.11, A' and B are interoperable and since, by hypothesis, there exists a successful disclosure sequence for A, B and res , there also exists a successful disclosure sequence for A', B and res . Thus, $\text{val}(T[A'/A]) = 1$.

Let B' be any $(k - 2)$ -cautious peer, and let T' be a transaction involving A and B' (i.e., $T' = \langle A, B', \text{res}', F_k \rangle$ or $T' = \langle B', A, \text{res}', F_k \rangle$). By Theorem 3.11, A' and B' are interoperable. So, $\text{val}(T') \leq \text{val}(T'[A'/A]) = 1$. Hence, A' dominates A in the class of $(k - 2)$ -cautious peers, which is a contradiction. ■

Now we have to provide operational guidelines for being interoperable with $(k - 2)$ -cautious clients. For this purpose, we introduce a relaxed notion of cautiousness called *weak cautiousness*. A peer is weakly cautious if it does not emit any vacuous messages when it “knows” that it could be useful to release some information. Then we prove that interoperability with $(k - 2)$ -cautious clients requires the strategy to be *weakly* $(k - 2)$ -cautious.

Definition 3.18 A peer A is *weakly* $(k - 2)$ -cautious if the following condition holds. For all Φ -transactions T involving A and some B , for all A -prefixes μ of $\text{nego}(T)$ having a vacuous tail whose length is $k - 2$ or greater, if there exists a $(k - 2)$ -cautious B' such that

- i. μ is a prefix of $\text{nego}(T[B'/B])$,
- ii. if $R_A(\mu)$ is vacuous then $T[B'/B]$ fails,

- iii. there exists a successful disclosure sequence for P_A , $P_{B'}$ and the initial request of T ,

then $\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu)$ implies $R_A(\mu) \not\subseteq \text{released}(\mu)$. ■

Condition (i) means that A cannot distinguish B from B' , as both peers would yield the sequence μ . Intuitively, A should check whether the other peer B *might* be a peer B' that might successfully complete the negotiation (by iii) but would react to a vacuous message with a failure (by ii). In that case, A should return a non-vacuous message if possible. The above definition can be reformulated in an equivalent but simpler way:

Definition 3.19 A peer A is *weakly $(k-2)$ -cautious* if the following condition holds. For all Φ -transactions T involving A and some B , and for all A -prefixes μ of $\text{nego}(T)$, if the following conditions hold:

- i. μ has a vacuous tail whose length is $k-2$ or greater,
- ii. if $R_A(\mu)$ is vacuous then T fails,
- iii. there exists a successful disclosure sequence for T ,

then $\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu)$ implies $R_A(\mu) \not\subseteq \text{released}(\mu)$. ■

Lemma 3.20 *Definitions 3.18 and 3.19 are equivalent.*

Theorem 3.21 *If A is Φ -interoperable with all $(k-2)$ -cautious peers, then A is weakly $(k-2)$ -cautious.*

Proof: By contradiction, assume that A is interoperable w.r.t. all $(k-2)$ -cautious peers, but A is *not* weakly $(k-2)$ -cautious. Then, there exists a transaction T involving A and some $(k-2)$ -cautious peer B s.t. $\text{nego}(T)$ has a prefix μ such that:

1. $\text{turn}(\mu) = A$,
2. μ has a vacuous tail of length at least $k-2$,
3. if $R_A(\mu)$ is vacuous then T fails,
4. $\text{unlocked}(P_A, \mu) \not\subseteq \text{released}(\mu)$ and $R_A(\mu) \subseteq \text{released}(\mu)$,
5. T has a successful disclosure sequence.

It follows that A cannot be interoperable w.r.t. $(k-2)$ -cautious peers because the transaction T is failed while it has a successful disclosure sequence. ■

Note that even if A is interoperable with all $(k-2)$ -cautious peers, still A might not be $(k-2)$ -cautious. Actually, A might not be $(k-1)$ -cautious either, as shown in the following example.

Example 3.22 Let $A' = (P_{A'}, R_{A'})$ be a peer s.t. $P_{A'} = \{c\}$ and $R_{A'}(\sigma) = \text{unlocked}(P_{A'}, \sigma)$, and $A = (P_A, R_A)$ be a peer s.t. $P_A = \{a, b\}$ and

$$R_A(\sigma) = \begin{cases} a & \text{if } \sigma_0 = \text{res} \\ \text{unlocked}(P_A, \sigma) & \text{otherwise.} \end{cases}$$

Moreover, let \emptyset unlocks a , \emptyset unlocks b , \emptyset unlocks c , $\{a\}$ unlocks res , $\{c\}$ unlocks res .

Note that for each ν s.t. $\nu_0 \neq \text{res}$, A is $(k-2)$ -cautious after ν and for each ν s.t. $\nu_0 = \text{res}$, A releases all it is needed to unlock res . It is easy to verify that A is iteroperable with all $(k-2)$ -cautious peers.

A is the client: Consider the transaction $T = \langle A, A', \text{res}, F_2 \rangle$. Then,

$$\text{nego}(T) = \{\text{res}\}, \{c\}, \{a\}, \{c\}, \{a\}$$

and A is not a $(k-1)$ -cautious peer.

A is the server: Consider the transaction $T_1 = \langle A', A, \text{res}, F_2 \rangle$. Then,

$$\text{nego}(T) = \{\text{res}\}, \{a\}, \{c\}, \{a\}, \{c\}$$

and A is not a $(k-2)$ -cautious peer. Now, if we consider the transaction $T_2 = \langle A', A, \text{res}, F_3 \rangle$ then

$$\text{nego}(T) = \{\text{res}\}, \{a\}, \{c\}, \{a\}, \{c\}, \{a\}$$

and A is not a $(k-1)$ -cautious peer.

Actually, weak cautiousness is necessary *and sufficient* to be interoperable with $(k-2)$ -cautious peers. As a corollary, weakly cautious peers are also cooperative with $(k-2)$ -cautious peers.

Theorem 3.23 *If A is weakly $(k-2)$ -cautious, then A is Φ -interoperable with all $(k-2)$ -cautious peers.*

Proof: By contradiction, let B be a $(k-2)$ -cautious peer and T be a transaction involving A and B , such that T has a successful disclosure sequence σ , but $\nu = \text{nego}(T)$ is failed, that is, ν has a vacuous tail $\nu_{n-k} \cdots \nu_n$ ($n = |\nu|$).

Let j be the least index such that $\sigma_j \not\subseteq \text{released}(\nu) = \text{released}(\nu_{<n-1})$. Since j is minimal, $\text{released}(\sigma_{<j}) \subseteq \text{released}(\nu_{<n-1})$, therefore all members p of $\sigma_j \setminus \text{released}(\nu)$ are unlocked in $\nu_{<n-1}$.

If such p were in P_B , then B should release some of them in ν_{n-1} or ν_n , because B is $(k-2)$ -cautious by assumption. Analogously, if those p were in P_A , then A should release some of them in ν_{n-1} or ν_n , because T satisfies the properties of Definition 3.19 and A is weakly $(k-2)$ -cautious. In both cases, some p would belong to $\text{released}(\nu)$ (a contradiction). ■

Corollary 3.24 *If A is weakly $(k-2)$ -cautious then A is Φ -cooperative w.r.t. $(k-2)$ -cautious peers.*

3.7 More on termination criteria

In this section we shall analyze the impact of different choices of k in the termination criterion F_k . This choice is important not only to help transactions succeed, but also to prevent “cunning” peers from forcing another peer to release all of its unlocked information.

In this section we consider the apparently non-problematic case of a negotiation between a monotonic, weakly $(k-2)$ -cautious server and a $(k-2)$ -cautious client (which are interoperable).

If we take F_1 as the termination criterion, then we obtain a negotiation framework Ψ_1 in which the release strategy of the client is always degenerate. More precisely, by Theorem 3.7, a client Ψ_1 -cooperative w.r.t. monotonic peers always releases all unlocked information at every step. Clearly, this property is highly undesirable for privacy-aware users.

Next assume $k > 1$. Are there any differences between even and odd k ? The answer is *yes*.

First, suppose k is odd. A server B can reply to a client’s request with vacuous messages to get as much information as possible. Even if a client A replies with more vacuous messages, after a sequence of $k-2$ vacuous messages A must release a non vacuous message (if any) because A is $(k-2)$ -cautious and $\text{turn}(\{\text{res}\} \emptyset^{k-2}) = A$. Peer B could iterate this process until A has something to release—in particular a sequence of $k-1$ empty messages tells B that A can release no more items, therefore B can safely detect when the time has come for releasing a new piece of information. The corresponding negotiation would be a sequence like:

$$\{\text{res}\} \emptyset^{k-2} m_1^A \emptyset^{k-2} m_2^A \dots \emptyset^{k-2} m_n^A \emptyset^{k-1} m_1^B \dots$$

where m_1^A, \dots, m_n^A are messages released by client A , \emptyset^{k-2} represents a $k-2$ long sequence of vacuous messages and m_1^B is a message released by server B .

Now suppose that k is even. In this case the server B cannot use the same trick to extract information from A because $\text{turn}(\{\text{res}\} \emptyset^{k-2}) = B$. Since B is cooperative w.r.t. $(k-2)$ -cautious peers, B must release a non-vacuous message (if at all possible). The corresponding negotiation would be a sequence like:

$$\{\text{res}\} \emptyset^{k-2} m_1^B \emptyset^{k-2} m_1^A \emptyset^{k-2} m_2^B \emptyset^{k-2} m_2^B \dots$$

until one of the peers really has no new piece of unlocked information (then a sequence \emptyset^{k-1} can be observed).

Therefore, even values for k are preferable. In order to minimize the number of vacuous messages (and make the protocol more efficient) the recommended choice is $k = 2$. Under the above assumptions on A and B , vacuous messages would always correspond to the actual lack of unreleased unlocked items.

4 Summary and perspectives

We have introduced an abstract framework that generalizes several concrete trust negotiation frameworks introduced in the literature, several of which have been actually implemented. Our results apply to all these concrete frameworks.

Through the notion of *cooperativeness* we modelled peers that are interested in making negotiations succeed. We studied the strategies driven by this unique goal. Unfortunately, our results prove that in general there are no guarantees of successful termination. Negotiations may fail just because confidential policies are not (entirely) disclosed, and because the strategy of the other peers are unknown. In technical terms, two cooperative peers are not necessarily *interoperable*. We proved much stronger results for the negotiations that involve mutually cooperative peers and monotonic servers (that model service providers of practical interest). Figure 1 summarizes these results, that are also discussed below.

To make a client A cooperative with monotonic peers, it is necessary and sufficient to program A ’s strategy in a $(k-2)$ -cautious way, that is, when failure is at most two steps away A should release at least one new unlocked item (if any). As an unexpected side effect, any two such clients are interoperable.

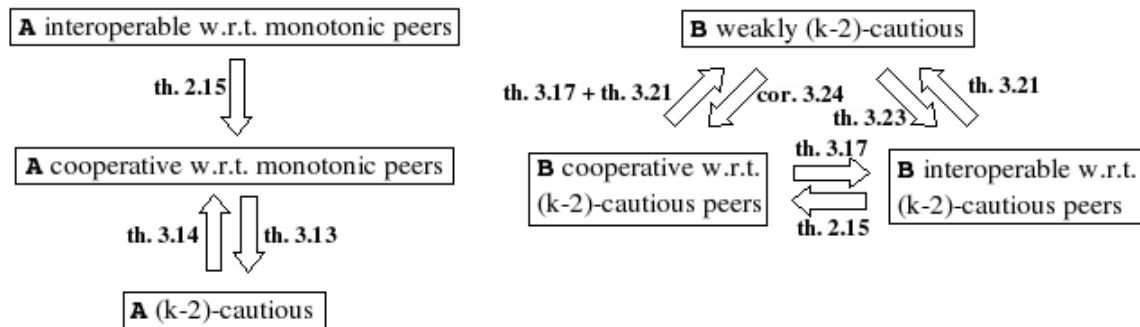
In the absence of further assumptions, cooperativeness does not entail interoperability with the servers. However, if a server is cooperative with $(k-2)$ -cautious clients, then the server is also interoperable with any such client. The necessary and sufficient operational criterion for being cooperative (and hence interoperable) is weak $(k-2)$ -cautiousness.

These results suggest to adopt frameworks where clients are $(k-2)$ -cautious and servers are monotonic and weakly $(k-2)$ -cautious. Finally, we argued that in these frameworks the appropriate failure condition is the occurrence of two consecutive vacuous messages.

While implementing $(k-2)$ -cautious strategies is trivial, programming weakly $(k-2)$ -cautious strategies may be tricky, and we suspect that with some policy languages \mathcal{L} weak cautiousness may be computationally hard. If this conjecture turned out to be true, then it might be preferable to adopt $(k-2)$ -cautious strategies also for servers, as an approximation of weak cautiousness. In this paper we have not investigated this problem because it requires more details on \mathcal{L} ’s syntax and semantics.

Our analysis should be extended to confidentiality issues (i.e., minimizing the amount and sensitivity of disclosed information), and to negotiations involving more than two peers.

Figure 1. Some of our main results



References

- [1] M. Y. Becker and P. Sewell. Cassandra: distributed access control policies with tunable expressiveness. In *5th IEEE International Workshop on Policies for Distributed Systems and Networks*, Yorktown Heights, June 2004.
- [2] P. Bonatti, C. Duma, N. Fuchs, W. Nejdl, D. Olmedilla, J. Peer, and N. Shahmehri. Semantic web policies - a discussion of requirements and research issues. In *Proc. of the European Semantic Web Conf. (ESWC 2006)*, pages 712–724, 2006.
- [3] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *CCS '00: Proceedings of the 7th ACM conference on computer and communications security*, pages 134–143. ACM Press, 2000.
- [4] P. A. Bonatti and D. Olmedilla. Driving and monitoring provisional trust negotiation with metapolicies. In *IEEE 6th International Workshop on Policies for Distributed Systems and Networks (POLICY 2005)*, Stockholm, Sweden, June 2005.
- [5] R. Gavriloaie, W. Nejdl, D. Olmedilla, K. Seamons, and M. Winslett. No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In *1st First European Semantic Web Symposium*, Heraklion, Greece, May 2004.
- [6] N. Li, J. Mitchell, and W. Winsborough. Design of a role-based trust-management framework. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 114. IEEE Computer Society, 2002.
- [7] K. Seamons, M. Winslett, T. Yu, B. Smith, E. Child, J. Jacobson, H. Mills, and L. Yu. Requirements for policy languages for trust negotiation. In *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02)*, Monterey, CA, June 2002. IEEE Computer Society.
- [8] S. Staab, B. Bhargava, L. Lilien, A. Rosenthal, M. Winslett, M. Sloman, T. S. Dillon, E. Chang, F. K. Hussain, W. Nejdl, D. Olmedilla, and V. Kashyap. The pudding of trust. *IEEE Intelligent Systems Journal*, 19(5):74–88, Sep/Oct 2004.
- [9] W. H. Winsborough, K. E. Seamons, and V. E. Jones. Automated trust negotiation. DARPA Information Survivability Conference and Exposition, IEEE Press, Jan 2000.
- [10] M. Winslett, N. Ching, V. Jones, and I. Slepchin. Assuring security and privacy for digital library transactions on the web: client and server security policies. In *IEEE ADL '97: Proceedings of the IEEE international forum on Research and technology advances in digital libraries*, pages 140–151. IEEE Computer Society, 1997.
- [11] M. Winslett, T. Yu, K. Seamons, A. Hess, J. Jarvis, B. Smith, and L. Yu. Negotiating Trust on the Web. *IEEE Internet Computing, special issue on trust management*, 6(6), Nov. 2002.
- [12] T. Yu, M. Winslett, and K. Seamons. Interoperable strategies in automated trust negotiation. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 146–155. ACM Press, 2001.
- [13] T. Yu, M. Winslett, and K. Seamons. Supporting Structured Credentials and Sensitive Policies through Interoperable Strategies in Automated Trust Negotiation. *ACM Transactions on Information and System Security*, 6(1), Feb. 2003.