Proceedings of the 2002 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 17–19 June 2002

Advancing Assurance for Secure Distributed Communications

Giampaolo Bella and Stefano Bistarelli

Abstract— **Securing distributed communications from malicious tampering is of capital importance. There exist a number of techniques addressing this issue but, to the best of our knowledge, an account for what Information Assurance means in this context is currently unavailable. A notion is advanced in this paper reducing Information Assurance for secure distributed communications to a threefold requirement for the protocols securing the communications. The protocols ought to be analysed *accurately*, *realistically* and *formally*. General considerations and specific examples are presented to enlighten the intuitive meaning of these terms exhaustively. This contribution aims at drawing attention to an important niche in computer security.**

I. Overview

Computer Security became a science (and an art) with the development of large-scale computer networks, that is in the late 1960s. Before those years, security merely meant to limit *physical* access to the sensitive nodes of the network. Technology progressed fast, thus the figure of a new expert became necessary, the *security architect*. We are not going to survey the history of computer security here, but a few observations are necessary to set the ground for our discussion on Information Assurance for secure distributed communications.

Security architects are concerned with security problems that require solutions, and the existing technology where to derive those solutions from. Until the 1980s, their decisions where only taken on the basis of *informal* reasoning. In practice, teams of architects used to brain-storm together for months or years in the hope to account for *all* possible strengths or weaknesses of a certain security measure. Only if ideas converged on the strengths would that measure be put in place.

But convergence of ideas very rarely derives from unanimous agreement. Indeed, enforcing certain measures in the real world may signify weakening others, which means that what was expected as a solution turns out to be in fact a tradeoff. An enormous number of examples can be made to support the claim that enforcing most security measures may still require acceptance of some level of risk, but we only sketch a couple here.

G. Bella is with Università di Catania, Dipartimento di Matematica e Informatica, Catania, Italy (giamp@dmi.unict.it)

S. Bistarelli is with CNR, Istituto di Informatica e Telematica, Pisa, Italy (stefano.bistarelli@iit.cnr.it) — Università "D'Annunzio", Dipartimento di Scienze, Pescara, Italy (bista@sci.unich.it)

"Security" installations. There exists empirical evidence that protecting a WWW server by allowing only SSH connections has decreased the number of external break-ins. Tolerable drawbacks are the cost of the new software, the time of installing SSH clients on all remote machines, and the efforts to create people's mentality to use the new software. But since the number of attacks has at times been too far from zero yet, many academic sites have decided to even restrict SSH access to principals within the local domain [1].

Analogous caution must be taken with security protocols, sets of of prescribed message exchanges between remote principals. Each protocol aims securing distributed communications by enforcing specific goals, such as confidentiality and authentication. Paulson has proved formally that the BULL recursive protocol meets its goals if encryption is *perfect*, that is if cryptanalysis is impossible [2]. This may have favoured the use of the protocol for some time, but Ryan and Schneider soon found an attack on the protocol if encryption is bit-wise exclusive or [3]. At installation time, the security architect would hopefully have the option of choosing a stronger encryption function, which in any case would be far from perfect in the sense of Paulson's. No boolean statement about the strengths of the installed software would be conceivable.

Semi-trusted code. Code downloaded from the Internet should not be entirely trusted. A number of viruses spread through the world as email attachments for example. A possible strategy to control the behaviour of the code and limit its potential damages is to execute it in a *sandbox*, as is the case of Java applets. The principal is granted fine-grained access control for each potentially dangerous call, so he can take informed decisions. But experience shows that to run code in a sandbox may result lengthy and tedious, so that the principal ends up giving many too permissions to the code anyway. Even more crucial is the problem of executing code from different origins, hence with different levels of trust, in the same runtime. Some levels may have to be lowered or decreased. Techniques based on inspection of the execution stack or on execution history are currently being developed [4].

The third Millennium seems to be favouring an increasing awareness about the issues sketched above, which sup-

ISBN 0-7803-9814-9/$10.00

port the claim that "Security is not a simple boolean predicate" [5]. Worse yet, complete security still seems out of reach or perhaps impossible. Modern security architects normally build their arguments on top of such postulates. As from the late 1980s they have also started to take into account insights derived from *formal* analyses, which were just beginning to be conceived, as opposed to the merely *informal* analyses they relied on before. Skepticism has gone up and down throughout the years, but the contribution of formal reasoning in general is nowadays unquestionable. This can be seen throughout nearly all niches of computer security, ranging from policies [6, 7] to protocols [8, 9].

Our purpose here is to provide an account for Information Assurance ("IA" in the following) in the field of secure distributed communications, which the literature appears to be lacking. We advance a tentative notion stating that

> *IA for secure distributed communications is reliance on security protocols that have undergone accurate, realistic and formal analysis.*

The general impossibility of making strong boolean claims in the field of security, which we have exemplified above, convinces that protocol analyses must be *accurate* and capture some notion of "level" of the goals achieved by a protocol. Likewise, protocol analyses must be *realistic* and admit that the present ratio cost/technology allows every Internet principal to be a potential attacker — each attacker is in a condition to exploit attacks mounted by other principals for his own sake. Also, security architects require formal assurance that a protocol meets its goals in the real world, which is intrinsically difficult because formal models somewhat idealise reality. Hence, an expressive *formalism* is needed.

The definition is purposely abstract so as to suscitate different views among different researchers, and hopefully give occasion to the international debate on a niche of computer security that at present deserves consideration. Our own views of the three provisos of accuracy, realism and formalism are given in the following sections respectively (§II, §III, §IV). The last section (§V) draws some conclusions.

II. Accuracy

We concentrate on the goals of confidentiality and authentication. Confidentiality of a message means that the message remains undisclosed to those not intended to learn it. Authentication of a principal means that we truly are communicating with that principal.

We observe that neither of these two properties is boolean in the real world: only certain *levels* of confidentiality or authentication are achieved in practice. Levels are known as a means to conduct reasoning, as is the case of Abadi's *types* [10], which means they are in fact meta-levels. By contrast, we are introducing object-levels. This section provides the basis for *accurate* protocol analyses, the first aspect of our notion of IA for secure distributed communications.

Confidentiality. To set about this goal, let us consider *session keys* for example. One or two of them are invented per each session (that is, execution) of a protocol, the reason being that each key is only meant to be used for a short lifetime. Clearly one such key is less sensitive information than a principal's password or PIN number, whose lifetimes usually are considerably long. That is to say that the acceptable level of confidentiality on a session key is lower than the acceptable level of confidentiality on a password. Most protocols follow this proviso. While session keys may be sent within message bodies on repeated occasions, passwords are only used as encryption keys typically once or twice. Indeed, sending a secret over the network exposes it to risks. The more the secret is used to form messages, the higher the risks that an attacker tampers with it — the longer the secret is on the network, the higher the risks it runs. Also, we may have chains of session keys, each encrypting the next one. The confidentiality levels of the keys decrease along the chain, as confidentiality of each key rests on confidentiality of all preceding keys. In practice, one would prefer to rely on cipher-texts sealed under the first key in the chain rather than under the last one. Kerberos [11] and the Yahalom protocol [12], for example, have dependency chains of length two. SET features an even longer chain [13].

Such a dependency chain is a didactic example, but we remark that *all* protocols exchange components some of which are more sensitive than others.

Once fixed criteria are established to manipulate the security levels, the protocol analyser could even compare the security level achieved by different protocols on sensitive message components. For example, the analyser could consider two different key-distribution protocols, such as Otway-Rees and Kerberos as they are presented by Burrows et al. [14], and study what confidentiality levels they achieve on the session key. This would give security architects deeper insights than current analyses do. As mentioned above, a secret runs risks of leaking that are proportional to the time it has been on the network. Therefore, one appropriate criterion to manipulate the security levels is to decrease the level associated to a message component every time that component is manipulated by any of the operations on messages (concatenation, splitting, encryption and decryption) — see §IV.

A message that has been disclosed to the attacker can be seen as having the minimum confidentiality level. This inspires a strategy to compare confidentiality attacks. For example, if we use Roman numbers, then leaking a message that was I-confidential reports a more significant attack than leaking a message that was V-confidential. To the best of our knowledge, such accuracy is missing to existing protocol analyses, but we find it an indispensable

ISBN 0-7803-9814-9/$10.00

prerequisite of appropriate IA for secure distributed communications.

Authentication. Analogous considerations apply to the authentication goal. Classifications of the different kinds of authentication exist [15, 16] but each of them should be studied in terms of the levels we introduce here.

Authentication (of whatever kind) of a principal is normally established by means of a message that expresses the principal's presence. The more the message is confidential, the stronger the achieved authentication. For example, sending Alice's identity on the network to Bob is the weakest form of authentication of Alice to Bob because message "Alice" is in fact public (all principals know each other's identities). It is however the most common form adopted at present, if we think that e-mail traffic routinely is in the clear. Clearly, a much stronger authentication Alice would achieve by sending Bob a cipher-text sealed under Alice's private key, which has a high confidentiality level. And some intermediate level of authentication Alice would get by sealing a cipher-text under some session key. The existence of a dependency chain generalises these observations.

It also is desirable to have a strategy to compare authentication attacks. A possible strategy is based on the following observation. If an attacker impersonates Alice with Bob, the significance of the attack is proportional to the confidentiality level of the message used by the attacker to impersonate Alice. For example, if the attacker gets hold of Alice's private signature key, he can mount a more significant authentication attack on Bob (and on any other principal) than if he gets hold of Alice's session key shared with Bob (in which case he could only cheat on that session).

III. Realism

Dolev and Yao's famous paper [17] has substantially influenced security protocol analyses during the entire last decade, regardless of whether they were informal or formal. Their contribution is essentially evidence that collusion of a number of principals to subvert a protocol is equivalent to the hostility of a single, powerful attacker. "Powerful" means the ability to monitor the entire network traffic, to break down messages by the conventional operations of splitting and decryption, to build up new messages by the conventional operations of concatenation and encryption, and to engage in unlimited protocol sessions. The only constraint is that the attacker cannot mount brute-force cryptanalysis attacks, hence he can only rely on the keys that become available by any interleaving of the four allowed operations.

These simplifications to the threat model characterise essentially all approaches to analysing security protocols formally. What has never been considered in this context is a threat model where each principal is malicious and is wishing to make his own profit in every way Dolev and Yao's attacker would. In other words, no principal colludes with any-one else, but they are all separate attackers, as we can see in the real world. The model is inspired to recent fair-exchange protocols [18]. The second aspect of our notion of IA for secure distributed communications, is the use of *realistic* threat models in the sense described above.

Here is one example of a previously unknown insight that our threat model allows us to get. Figure 1 presents the simple protocol due to Needham and Schroeder [19], a milestone in the field. This is the variant that relies on asymmetric cryptography, that is each principal owns a private key and a corresponding public key. A *nonce* is a "number that is used only once" [19]. The protocol assumes that principals can invent *truly-random* nonces, so that, given a nonce N invented by an principal P, the probability that principals other than P guess N is negligible.

$$\begin{array}{lll} 1. & A \rightarrow B & : \{\!|Na, A|\!\}_{Kb} \\ 2. & B \rightarrow A & : \{\!|Na, Nb|\!\}_{Ka} \\ 3. & A \rightarrow B & : \{\!|Nb|\!\}_{Kb} \end{array}$$

Fig. 1. The asymmetric Needham-Schroeder protocol

The first step sees an *initiator* A initiate the protocol with a *responder* B. Principal A invents a nonce Na and encrypts it along with her identity under B's public key. Upon reception of that message, B decrypts it and extracts A's nonce. Then, he invents a nonce Nb and encrypts it along with Na under A's public key. When A receives message 2, she extract Nb and sends it back to B, encrypted under his public key.

The goal of the protocol is authentication: at completion of a protocol session initiated by A with B, A should get evidence to have communicated with B and, likewise, B should get evidence to have communicated with A. We emphasise how confidentiality of the nonces is here used to achieve authentication. Indeed, upon reception of Na inside message 2, A would conclude that she is interacting with B, the only principal who could retrieve Na from message 1, since Na is a truly-random nonce and encryption is perfect. In the same fashion, when B receives Nb inside message 3, he would conclude that A was at the other end of the network because Nb must have been obtained from message 2, and no-one but A could perform this operation. However, Lowe shows that this protocol is flawed [20] by exhibiting the attack we present in figure 2.

Notice that C could be a registered principal of the network, so no-one could suspect his tampering. Since A initiates with C, she encrypts her nonce and her identity under C's public key. Once obtained these data, C initiates another session (indicated by the primes) with another principal B, quoting A's data rather than his own. From this

ISBN 0-7803-9814-9/$10.00

$$
\begin{array}{lll}
1. & A \rightarrow C & : \{\!|Na, A|\!\}_{Kc} \\
1'. & C \rightarrow B & : \{\!|Na, A|\!\}_{Kb} \\
2'. & B \rightarrow A & : \{\!|Na, Nb|\!\}_{Ka} \\
2. & C \rightarrow A & : \{\!|Na, Nb|\!\}_{Ka} \\
3. & A \rightarrow C & : \{\!|Nb|\!\}_{Kc} \\
3'. & C \rightarrow B & : \{\!|Nb|\!\}_{Kb}
\end{array}
$$

Fig. 2. Lowe's attack to the Needham-Schroeder Protocol

message, B deduces that A is trying to communicate with him. Therefore, B replies to A, quoting her nonce and his own, Nb. Since the entire network is under C's control, C intercepts this message before it is delivered to A but cannot decrypt it because encryption is perfect. So, C forwards it to A. The message is of the form that A was expecting, hence A extracts Nb and sends it to the principal with whom she had initiated the first session, C. This hinders the confidentiality of Nb, so C can use it to complete the session with B by issuing message 3', which is of the form that B was expecting.

As a result, B believes to have communicated with A, while A was in fact communicating with C. In other words, C impersonates A with B: the protocol has ultimately failed achieve authentication because it has failed to keep Nb confidential. Lowe observes that this may have drastic consequences, such as the following. If B is a bank, C can steal money from A's account as in figure 3. The bank B would honour the request believing it came from the account holder A.

$$
C \rightarrow B : \quad \{\!|Na, Nb, \text{“Transfer } \pounds 1000 \text{ from } A\text{'s account to } C\text{'s”}|\!\}
$$

Fig. 3. Lowe's fraud to bank B

This terminates Lowe's study, which is sound within Dolev and Yao's threat model. We studied the protocol within our own threat model and highlighted an *indeliberate attack* [21] whereby B learns A's nonce Na, which is meant for use with C, not with B. "Indeliberate" here means that B obtains the nonce without any deliberate action. Still, this contrasts the protocol policy and indeed may have another drastic consequence. If A is a bank, B can steal money from C's account as in figure 4. The bank A would honour the request believing it came from the account holder C.

The threat model given here allows *retaliation* of B against C, a novel concept that has been recently developed [22]. It appears realistic at present.

$$
B \rightarrow A : \quad \{\!|Na, Nb, \text{“Transfer } \pounds 1000 \text{from } C\text{'s account to } B\text{'s”}|\!\}
$$

Fig. 4. Our fraud to bank A

IV. Formalism

At present, it is well known that formal analyses of security protocols certainly get more impact than informal analyses. The former have been conducted using a variety of approaches, ranging from *state enumeration* [9], to *provable security* [23], to induction [24]. These efforts have led to the discovery of a number protocol attacks (and of the corresponding patches), or to the formal establishment that certain goals are achieved.

However, we observe that the exhibition of an attack raises more interest among the security architects than the exhibition of a proof that a goal is met. While it is easy to verify that the former can take place, it is less easy to believe that a formal proof would still hold in the real world. Such skepticism is motivated by the idealised models within which any formal proof is conducted. Although it is virtually impossible to bridge the gap between a formal model and the real world, following our second requirement (§III) certainly is an improvement. Reluctance towards formal proofs that goals are met also comes from the nature of the offered insights. These are firm boolean claims of the form "session key K is confidential" or "principal A authenticates principal B", while security architects in practice rely on levels of those goals, whose importance was stated above by our first requirement (§II).

The third aspect of our notion of IA for secure distributed communications is therefore *formal* analysis, which, as remarked above, is severely limited on its own, but can be powered with the other two requirements. It was not obvious in the beginning what approach to protocol analysis could embed all three requirements, but Constraint Solving soon seemed to be an appropriate candidate [25]. We present here only the basics of our approach, which is at the same time accurate, realistic and formal. The complete description can be found elsewhere [26]. It should be remarked that ours is *one* possible approach that embeds all three features, but certainly others may be taken. We expect that some existing formal approaches can be extended to accommodate our first two requirements (§II, §III).

A. Basics of Soft Constraint Programming

Informally speaking, given a set of variables $\mathcal{V}$ and a set of domain values $\mathcal{D}$, a *constraint* is a law that associates n-tuples of domain elements to n-tuples of variables. A *soft constraint* is a constraint where each association of its variables has an associated value from a partially ordered

ISBN 0-7803-9814-9/$10.00

set $\mathcal{A}$. On this set, two operations are defined that allow for combination, $\times$, and comparison, $+$. If $\mathbf{0}$ is the unit element of $+$, and $\mathbf{1}$ is the unit element of $\times$, we can require appropriate properties on the two operations so that the tuple $\langle \mathcal{A}, +, \times, \mathbf{0}, \mathbf{1} \rangle$ is a c-semiring [27].

Let us consider the relation $\leq_S$ over A such that $a \leq_S b$ iff $a + b = b$. Then it is possible to prove that:

- $\leq_S$ is a partial order;
- $+$ and $\times$ are monotone on $\leq_S$;
- $\mathbf{0}$ is its minimum and $\mathbf{1}$ its maximum;
- $\langle A, \leq_S \rangle$ is a complete lattice and, for all $a, b \in A$, $a + b = lub(a, b)$.

The relation $\leq_S$ gives us a way to compare (some of the) tuples of values and constraints. In fact, when we have $a \leq_S b$, we will say that b *is better than* a. Below, $a \leq_S b$ will be often indicated by $\leq$.

A *constraint system* is a tuple $CS = \langle \mathcal{S}, \mathcal{D}, \mathcal{V} \rangle$ where $\mathcal{S}$ is a c-semiring, $\mathcal{D}$ is a finite set (the domain of the variables) and $\mathcal{V}$ is an ordered set of variables. Given a semiring $\mathcal{S} = \langle A, +, \times, \mathbf{0}, \mathbf{1} \rangle$ and a constraint system $CS = \langle \mathcal{S}, \mathcal{D}, \mathcal{V} \rangle$, a *constraint* is a pair $\langle def, con \rangle$ where $con \subseteq \mathcal{V}$ and $def : \mathcal{D}^{|con|} \rightarrow A$. Therefore, a constraint specifies a set of variables (the ones in *con*), and assigns to each tuple of values of these variables an element of the semiring.

A *soft constraint satisfaction problem* (SCSP) is a pair $\langle C, con \rangle$ where $con \subseteq \mathcal{V}$ and C is a set of constraints: *con* is the set of variables of interest for the constraint set C, which however may concern also variables not in *con*.

Combining and projecting soft constraints. Given two constraints $c_1 = \langle def_1, con_1 \rangle$ and $c_2 = \langle def_2, con_2 \rangle$, their *combination* $c_1 \otimes c_2$ is the constraint $\langle def, con \rangle$ defined by $con = con_1 \cup con_2$ and $def(t) = def_1(t \downarrow_{con_1}^{con}) \times def_2(t \downarrow_{con_2}^{con})$, where $t \downarrow_Y^X$ denotes the tuple of values over the variables in Y, obtained by projecting tuple t from X to Y. In words, combining two constraints means building a new constraint involving all the variables of the original ones, and which associates to each tuple of domain values for such variables a semiring element which is obtained by multiplying the elements associated by the original constraints to the appropriate subtuples.

Given a constraint $c = \langle def, con \rangle$ and a subset I of $\mathcal{V}$, the *projection* of c over I, written $c \Downarrow_I$ is the constraint $\langle def', con' \rangle$ where $con' = con \cap I$ and $def'(t') = \sum_{t/t \downarrow_{I \cap con}^{con} = t'} def(t)$. Informally, projecting means eliminating some variables. This is done by associating to each tuple over the remaining variables a semiring element which is the sum of the elements associated by the original constraint to all the extensions of this tuple over the eliminated variables.

Solutions. The *solution* of an SCSP problem $\mathsf{P} = \langle C, con \rangle$ is the constraint $Sol(\mathsf{P}) = (\bigotimes C) \Downarrow_{con}$. That is, we combine all constraints, and then project over the variables in *con*. In this way we get the constraint over *con* which is "induced" by the entire SCSP.

B. Soft Constraint Programming to analysing security protocols

We define the *security semiring* to specify each principal's trust on the confidentiality of each message, that is each principal's *security level* on each message. The security levels form the career set L of the security semiring.

$$
\begin{aligned}
L = \{ & unknown && \equiv traded_{-1}, \\
& private && \equiv traded_0, \\
& traded_1, && \\
& traded_2, && \\
& \ldots, && \\
& traded_n, && \\
& public && \equiv traded_{n+1} \}
\end{aligned}
$$

They range from the most secure (highest) level *unknown* (double named as traded$_{-1}$) to the least secure (lowest) level *public* (double named as traded$_{n+1}$). Intuitively, if A's security level on m is *unknown*, then no principal (included A) knows m according to A, and, if A's security level on m is *public*, then all principals potentially know m according to A. The lower A's security level on m, the higher the number of principals knowing m according to A. For simplicity, we state no relation between the granularity of the security levels and the number of principals. We define $+_{sec}$ and$\times_{sec}$ by the following axioms.

Ax. 1: $traded_i \ +_{sec} \ traded_j \ = \ traded_{min(i,j)}$
Ax. 2: $traded_i \ \times_{sec} \ traded_j \ = \ traded_{max(i,j)}$

The structure
$\mathcal{S}_{sec} = \langle L, +_{sec}, \times_{sec}, public, unknown \rangle$ can be easily proved to be a c-semiring.

Using the security semiring, we define the *network constraint system*, which represents the computer network on which the security protocols can be executed. It does not depend on any specific protocol. It is expressed as $CS_n = \langle \mathcal{S}_{sec}, \mathcal{D}, \mathcal{V} \rangle$ where:

- $\mathcal{S}_{sec}$ is the security semiring just mentioned;
- $\mathcal{V}$ is a bounded set of variables, each standing for a principal;
- $\mathcal{D}$ is a bounded set of values including the empty message $\{\!|\}$ and all atomic messages, as well as all messages recursively obtained by concatenation and encryption.

The development of the principals' security levels from manipulation of the messages seen during the protocol sessions can be formalised as a *security entailment*, which is an entailment relation between constraints [21]. The relation is defined by four rules, one for each operation on the messages (splitting, decryption, concatenation and encryption). In brief, every time a principal invents a secret message, the principal's security level on the message decreases from *unknown* to *private*; every time the message is sent on the network the secret level of the message is decreased (for example from *private* to $traded_1$, from

ISBN 0-7803-9814-9/$10.00 ©2002 IEEE

$traded_1$ to $traded_2$, etc.) to represent exposure to the network risks. This influences the principal's security levels on all messages that feature that secret, whose new (decreased) levels are computed by entailment. For example, encryption and concatenation build up new messages from known ones. The new messages must not get a worse security level than the known ones have. So, the corresponding rules choose the better of the given levels [26]. Precisely, if messages m_1 and m_2 have security levels v_1 and v_2 respectively, then the encrypted message $\{\!|m_1|\!\}_{m_2}$ and the compound message $\{\!|m_1, m_2|\!\}$ get a new level that is the better of v_1 and v_2, that is $v_1 +_{sec} v_2$.

At this stage, given a specific protocol, we represent the policy that accompanies the protocol as an SCSP called the *policy SCSP*. It formalises all admissible network configurations arising from the protocol execution as prescribed by the protocol designers. Therefore, any interleaving of protocol sessions in which no principal has acted maliciously is represented in the policy SCSP for the given protocol. The exact construction is done algorithmically, but is irrelevant to our discussion. Then, a particular network configuration arising from the protocol execution in the real world can be represented as another SCSP, an *imputable SCSP*. We have designed another algorithm for this task. There exists one such SCSP per each possible network configuration under the given protocol, while there exists one policy SCSP per each protocol [26].

Given a security level l, we use *l-confidentiality* and *l-authentication* to formally capture the level of achievement of the goal. In case of confidentiality of a message for a principal in an SCSP, that level is the principal's security level on the message. It is computed by calculating the solution of the SCSP, projecting it on the principal and evaluating it on the message.

Definition 1 (l-confidentiality) Given an SCSP p, let $Sol(\mathsf{p}) \Downarrow_{\{A\}} = \langle def_A, \{A\}\rangle$; *$l$-confidentiality of m for A in* p holds iff $def_A(m) = l$.

By comparing the solutions of the policy and the given imputable SCSPs we can formally define a confidentiality attack.

Definition 2 (Confidentiality attack)
Given the policy SCSP P for a given protocol, and an imputable SCSP p for the same protocol, that there is a *confidentiality attack by A on m in* p iff l-confidentiality of m in P for A holds, l'-confidentiality of m in p for A holds, and $l' < l$.

Therefore, if $Sol(\mathsf{P}) \Downarrow_{\{A\}} = \langle Def_A, \{A\}\rangle$, there is a confidentiality attack by A on m in p iff $def_A(m) < Def_A(m)$. Attacks can be realistically compared: the more an attack lowers a security level allowed by the policy SCSP, the worse that attack.

We exemplify this treatment on the Needham-Schroeder protocol seen above (§III). Figure 5 presents the fragment of policy SCSP for the protocol pertaining to a single session between principals A and B. Notice the unary constraints formalising each principal's security levels prior to the beginning of any protocol session, and the binary constraints each formalising a session step. In particular, while A's security level on her nonce Na was initially *private* before any session began, it is lowered to $traded_1$ by entailment as soon as A invents it and sends it off in step 2 of the protocol. Similarly, B's security level on Nb is $traded_2$ though it was originally *private*. The figure omits all details that are irrelevant to the session. For example all other principals' security levels on Na and Nb are *unknown* because the policy prescribes that no-one acts maliciously.

Figure 6 formalises the network configuration defined by Lowe's attack. The solution of this SCSP projected on variable C is a constraint that associates security level $traded_4$ to the nonce Nb. Following definition 1, Nb is $traded_4$-confidential for C in this SCSP. Hence, by definition 2, there is a confidentiality attack by C on Nb in this problem, because Nb got level *unknown* in the policy SCSP.

The problem solution projected on variable B associates security level $traded_2$ to the nonce Na, which instead got level *unknown* in the policy SCSP. This signifies that B has learnt a nonce that he was not allowed to learn by policy, hence there is an indeliberate confidentiality attack by B on Na. Notice that the two attacks (the deliberate [20] and the indeliberate [21]) are uniformly formalised.

Our approach is accurate, realistic and formal. It is amenable to mechanization by model checking if we bound all quantities, hence the possible network configurations are finite [28].

V. Conclusions

We have laid the ground towards the development of a definition of Information Assurance for secure distributed communications. We require reliance on security protocols previously analysed accurately, realistically and formally.

"Accurately" means that the protocol goals should not be considered mere boolean properties because security never is a boolean feature. By contrast, we advocate reasoning about levels of confidentiality or authentication. "Realistically" means that the model underlying the analysis should exceed the limits of the classical Dolev and Yao's model. We showed how this highlights another consequence of Lowe's attack on the popular asymmetric Needham-Schroeder protocol. "Formally" means that the analysis should be conducted within a formal framework.

The literature seems to be missing an approach to protocol analysis that embeds all the three features, so we have sketched a new approach based on Constraint Solving. Admittedly, some details had to be hidden from the presentation of the approach. But rather than convincing that this is the best approach, our purpose was to convince the reader that the three requirements we set towards Information Assurance for secure distributed communications can

ISBN 0-7803-9814-9/$10.00

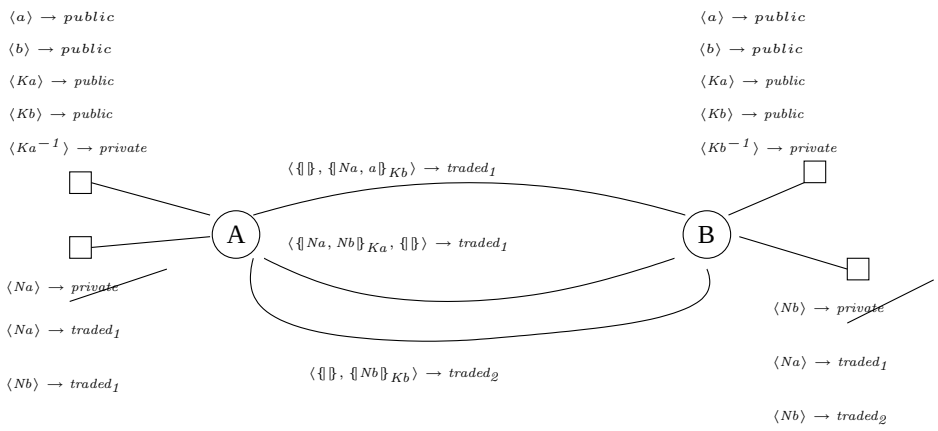

Fig. 5. Fragment of the policy SCSP for the Needham-Schroeder protocol

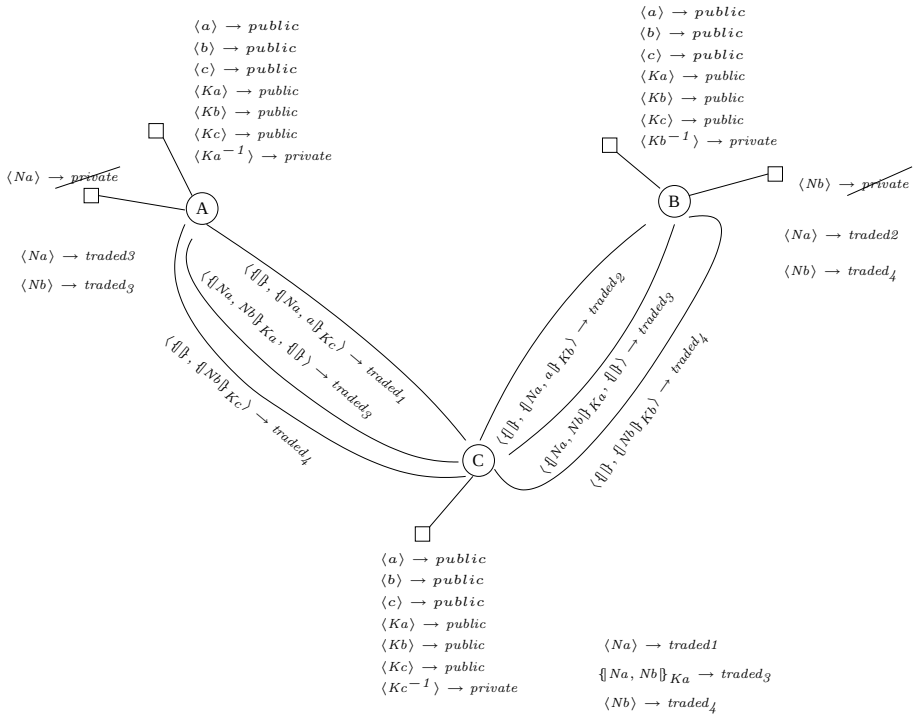

Fig. 6. Fragment of the Imputable SCSP corresponding to Lowe's attack

coexist together within a single approach to analysing the underlying security protocols.

Acknowledgements

We are grateful to Simon Foley, Michael Leuschel and Fabio Massacci for useful discussions on the topic of this paper.

References

[1] W. Cheswick, S. M. Bellovin, and A. D. Rubin, *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, 2004.

[2] L. C. Paulson, "Mechanized Proofs for a Recursive Authentication Protocol," pp. 84–95, 1997.

[3] P. Y. A. Ryan and S. A. Schneider, "An Attack on a Recursive Authentication Protocol: A Cautionary Tale," in *Information Processing Letters 65*, 1998.

[4] M. Abadi and C. Fournet, "Mobile Values, New Names, and Secure Communication," pp. 104–115, 2001.

[5] R. Anderson, "Why Cryptosystems Fail," pp. 217–227, 1993.

[6] L. Cholvy and F. Cuppens, "Analyzing Consistency of Security Policies," 1997.

[7] J. Y. Halpern and V. Weissman, "Using First-Order Logic to Reason about Policies," 2003.

[8] F. J. T. Fábrega, J. C. Herzog, and J. D. Guttman, "Strand Spaces: Why is a Security Protocol Correct?," 1998.

[9] P. Y. A. Ryan and S. A. Schneider, *The Modelling and Analysis of Security Protocols: the CSP Approach*. 2000.

[10] M. Abadi, "Secrecy by Typing in Security Protocols," *Journal of the ACM*, vol. 46, no. 5, pp. 749–786, 1999.

[11] G. Bella and L. C. Paulson, "Kerberos Version IV: Inductive

Analysis of the Secrecy Goals," LNCS 1485, pp. 361–375, 1998.
[12] L. C. Paulson, "Relations between Secrets: two Formal Analyses of the Yahalom Protocol," 2000. In press.
[13] G. Bella, F. Massacci, and L. C. Paulson, "Verifying the SET Registration Protocols," *IEEE Journal of Selected Areas in Communications*, vol. 21, no. 1, pp. 77–87, 2003.
[14] M. Burrows, M. Abadi, and R. M. Needham, "A Logic of Authentication," *Proc. of the Royal Society of London*, vol. 426, pp. 233–271, 1989.
[15] D. Gollmann, "What do we mean by Entity Authentication?," pp. 46–54, 1996.
[16] G. Lowe, "A Hierarchy of Authentication Specifications," pp. 31–43, 1997.
[17] D. Dolev and A. Yao, "On the security of public-key protocols," *IEEE Transactions on Information Theory*, vol. 2, no. 29, 1983.
[18] V. Shmatikov and J. C. Mitchell, "Analysis of a fair exchange protocol," in *Network and Distributed System Security Symposium (NDSS-00)*, pp. 119–128, 2000.
[19] R. M. Needham and S. M. D., "Using Encryption for Authentication in large Networks of Computers," *Communications ACM*, vol. 21, no. 12, pp. 993–999, 1978.
[20] G. Lowe, "An Attack on the Needham-Schroeder Public-Key Authentication Protocol," *Information Processing Letters*, vol. 56, no. 3, pp. 131–133, 1995.
[21] G. Bella and S. Bistarelli, "Confidentiality levels and deliberate/indeliberate protocol attacks," in *Proc. 10th International Workshop on Security Protocols Cambridge, UK*, LNCS, Springer, 2002. To appear.
[22] G. Bella, S. Bistarelli, and F. Massacci, "A protocol's life after attacks ... ," in *Proc. 11th International Workshop on Security Protocols Cambridge, UK*, LNCS, Springer, 2003. To appear.
[23] M. Bellare and P. Rogaway, "Provably Secure Session Key Distribution — the Three Party Case," pp. 57–66, 1995.
[24] L. C. Paulson, "The Inductive Approach to Verifying Cryptographic Protocols," vol. 6, pp. 85–128, 1998.
[25] S. Bistarelli, *Semirings for Soft Constraint Solving and Programming*, vol. 2962 of *LNCS*. Springer, 2004.
[26] G. Bella and S. Bistarelli, "A multilevel analysis of security protocol goals," *Theory and Practice of Logic Programming (TPLP)*, 2004. To appear.
[27] S. Bistarelli, U. Montanari, and F. Rossi, "Semiring-based Constraint Solving and Optimization," *Journal of the ACM*, pp. 201–236, 1997.
[28] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov, "Undecidability of bounded security protocols," in *Proc. FMSP'99*, 1999.

ISBN 0-7803-9814-9/$10.00