Paper

Analysis of cryptographic protocols using logics of belief: an overview

David Monniaux

Abstract — When designing a cryptographic protocol or explaining it, one often uses arguments such as "since this message was signed by machine *B*, machine *A* can be sure it came from *B*" in informal proofs justifying how the protocol works. Since it is, in such informal proofs, often easy to overlook an essential assumption, such as a trust relation or the belief that a message is not a replay from a previous session, it seems desirable to write such proofs in a formal system. While such logics do not replace the recent techniques of automatic proofs of safety properties, they help in pointing the weaknesses of the system. In this paper, we present briefly the BAN (Burrows-Abadi-Needham) formal system [10, 11] as well as some derivative. We show how to prove some properties of a simple protocol, as well as detecting undesirable assumptions. We then explain how the manual search for proofs can be made automatic. Finally, we explain how the lack of proper semantics can be a bit worrying.

Keywords — cryptographic protocols, logics of belief, BAN, GNY, decidability.

1. Why logics of belief?

Cryptographic protocols are usually specified as sequences of messages in the following kind of format:

Needham-Schroeder shared-keys protocol [10, 17, 24]

$$
\begin{array}{lll}
1. & P \rightarrow S & : P, Q, N_p \\
2. & S \rightarrow P & : \left\{N_p, Q, K_{pq}, \{K_{pq}, P\}_{K_{qs}}\right\}_{K_{ps}} \\
3. & P \rightarrow Q & : \{K_{pq}, P\}_{K_{qs}} \\
4. & Q \rightarrow P & : \{N_q\}_{K_{pq}} \\
5. & P \rightarrow Q & : \{N_q - 1\}_{K_{pq}}
\end{array}
$$

where *S*, *P*, *Q* are machines or *principals*, or rather roles for machines in this protocol. *S*, as usual, designates a server.

N_p and N_q are *nonces* [21, §10.5]; these are random numbers (in this case, chosen respectively by *P* and *Q*) used to prevent *replay attacks*. Such attacks consist in an intruder replaying parts of messages recorded during previous sessions. The usual use of nonces is that the principals check that the values in certain encrypted message fields correspond to the correct values of the nonces for this session; discrepancies, arising from instance from messages recorded during previous sessions, get detected, preventing the principals from accepting those messages in a successful run of the protocol. From the point of view of protocol analysis, nonces are treated as being distinct from any other data used in the protocol. A related concept is that of *confounders* [21, §10.5], random numbers incorporated into messages to foil chosen plaintext attacks on public-key ciphers.

K_{xy} is a generic notation for a key shared between *x* and *y*. The goal of this protocol is to allow *P* and *Q* to agree on a shared communication key K_{pq}; for this, on the one hand, *P* and *Q* call a trusted server *S* which generates the key during the execution of the protocol; on the other hand, *S* communicates with *P* and *Q* using shared keys, K_{ps} and K_{qs} respectively, which are supposed to be known initially by the concerned parties.

The above description is a bit ambiguous, since it uses the same name (say, *K*) both for data that a principal generates by itself and for data that a principal receives from outside. For instance, in message 1, N_p is generated by *P* and thus treated by *P* as a known constant, but is received by *S* and thus treated by *S* as a variable. It can nevertheless be made unambiguous by distinguishing those two uses. From such an explicit description we can derive a semantics; that is, we describe in a mathematical way the actions of the principal. We also assume that we are in the Dolev-Yao model [13]: the cryptography is perfect, the intruder has full control of the network and can listen to, cancel and forge messages. Various analysis techniques, some of which considerably automated [5, 6, 8, 20, 26, 29, 32, and many others], have been applied to this model to obtain proofs of certain properties, and more particularly secrecy.

For all the successes of the Dolev-Yao model, using it to plan the design of a protocol is unnatural for a human. People do not design protocols by enumerating all the actions that could take place; they rather think of higher-order concepts such as "secret key only known to *A* and *B* and used to communicate between them" and form inferences such as "if a message arrives encrypted with a key known only to me and machine *M*, and I did not send it originally, then it must have been sent by *M*"[1]. Such reasoning is informal, which can be seen as a weakness. For this reason, some *logics of belief*, aiming at formalizing such inferences, have been proposed. The first of these was the so-called BAN logic from Burrows, Abadi and Needham [10, 11], which was followed by more expressive and elaborate extensions such as GNY (Gong, Needham and Yahalom [16, 17]), (Syverson and van Oorschot [33, 34]) and CKT5 [9]. One limitation of these logics is the need to annotate the protocols with logical assertions that are assumed to represent the intent of the sender of the message, as well as logical assumptions on the secrecy or freshness

[1] See [36] for a long discussion on such issues.

of certain pieces of information. Also, they cannot verify secrecy; in fact, *they make the implicit assumption that secrets are protected* [25].

BAN and subsequent logics are *modal logics of belief*; they deal with the beliefs that the principals can hold about their environment, for instance, about the distribution of the shared keys. That notion of "beliefs" is to be understood as the beliefs that a human playing the role of the principal may reasonably hold; "sensible" rules of deduction will be provided in the definition of the logic. As we will explain later (Section 4), it is difficult to provide a more precise semantics.

2. A short presentation of BAN and GNY logics

2.1. BAN logic

BAN logic [10, 11] is a many-sorted modal logic, which distinguishes between several sorts of objects: principals, encryption keys, nonces, and formulas, or statements. The first three sorts of objects have already been seen 1; the last sort is defined by the following syntax (taken from [10, pp. 4–5]; we left out two less used constructs):[2]

- $P \mid\equiv X$: *P believes X*.
- $P \triangleleft X$: *P sees X*. P initially knew or has received the message X and can read and repeat it.
- $P \mid\sim X$: *P once said X*. P has at one time sent a message containing the statement X. It is not known whether the message was sent long ago or during the current session of the protocol, but it is known that P believed X when it sent the messages.
- $P \Mapsto X$: *P has jurisdiction* over X and should be trusted on this matter. For instance, key distribution servers will be trusted for statements pertaining to keys.
- $\sharp(X)$: X is *fresh*; that is, X has not been sent in a message at any time before the current run of the protocol. This is usually true for nonces; $\sharp(X)$ will then be used as a complement to $P \mid\sim X$ to establish that a message from P is really about the current session and is not some old recorded message used by the intruder in a replay attack.
- $P \overset{K}{\leftrightarrow} Q$: P and Q may use the shared key K to communicate. The key K is good, in that it will never be discovered by any principal except P or Q, or a principal trusted by either P or Q. Note that we make here the assumption that secrets are protected. This symbol is commutative, i.e. $P \overset{K}{\leftrightarrow} Q$ is equivalent to $Q \overset{K}{\leftrightarrow} P$.
- $\overset{+K}{\mapsto} P$: P has K as a public key. The matching secret key (the inverse of K, denoted $-K$) will never be discovered by any principal except P, or a principal trusted by P.
- $\{X\}_K$: This represents the formula X encrypted under the key K. A weird point of BAN-like logics is that they consider that one can encrypt beliefs represented in formula. We shall now see why.

  Since messages are considered from the point of view of their meaning, a message K_{pq} conveying a key to be used between P and Q is represented in the logic as $P \overset{K_{pq}}{\longleftrightarrow} Q$. Since the key is generally encrypted so as to not being divulged to the intruder, the actually transmitted message is encrypted, for instance $\left\{K_{pq}\right\}_{K_{qs}}$. The corresponding formula is $\left\{P \overset{K_{pq}}{\longleftrightarrow} Q\right\}_{K_{qs}}$.
- (X,Y) represents the pair, or concatenation, of X and Y. Note that this symbol will be treated as commutative and associative.

We shall now see the deduction rules of BAN logic. A *deduction rule* is simply a set of *premises*, or *hypotheses* $\mathscr{H}_1, \ldots, \mathscr{H}_n$ and a *conclusion* $\mathscr{C}$ written as formulas with variables. Those variables stand for any formula, principal or nonce. We shall write such a rule as follows:

$$\frac{\mathscr{H}_1 \cdots \mathscr{H}_n}{\mathscr{C}}.$$

Such rules allow writing proofs as trees, whose leaves are the assumptions of the protocol or some already proved intermediary results and whose nodes are applications of the rules (see Figs. 1 and 2 for examples of somewhat complex proof trees). The notation

$$\frac{\begin{array}{c}\vdots\,\alpha_1\\ \mathscr{H}_1\end{array} \quad \cdots \quad \begin{array}{c}\vdots\,\alpha_n\\ \mathscr{H}_n\end{array}}{\mathscr{C}}$$

means that α_i designates the branch of the proof tree whose root is $\mathscr{H}_i$. In our list of the rules for BAN logic, we shall use this notation to identify some premises in some rules, the use of which will be explained in Subsection 3.2.

- The *message-meaning* rules concern the interpretation of messages authenticated by encryption using a shared or private key:

$$\frac{\begin{array}{c}\vdots\,p_1\\ P \mid\equiv P \overset{K}{\leftrightarrow} Q\end{array} \quad \begin{array}{c}\vdots\,p_2\\ P \triangleleft \{X\}_K\end{array}}{P \mid\equiv Q \mid\sim X}\ \mathsf{MM1}$$

  The reasoning behind that rule is that if a key K is shared between two principals P and Q and is kept secret, if P sees a message encrypted with K, then it

[2] We use the original notation from the authors [10], who later preferred a more readable, albeit more verbose, notation [11]. We unfortunately cannot use this latter notation due to width constraints.

can assume it comes from Q. An additional (and easy to overlook) assumption is that the message should not have originally come from P. Burrows, Abadi and Needham justify this by explaining that $\{X\}_K$ is actually an abbreviation for $\{X\}_K$ *from* P, meaning that the encryption was done by P. It is assumed that each principal can recognize messages that it encrypted itself and ignore them. The message meaning rule can then be rewritten as:

$$\frac{P \mid\equiv P \overset{K}{\leftrightarrow} Q \qquad P \triangleleft \{X\}_K \ \mathit{from} \neq P}{P \mid\equiv Q \mid\!\sim X}\ \mathsf{MM1}.$$

This is a bit uneasy. GNY logic (Subsection 2.3) introduces a symbol $\star$, meaning *not originated here*, which makes such considerations internal to the logic:

$$\frac{\overset{\vdots\ p_1}{P \mid\equiv \overset{+K}{\mapsto} Q} \quad \overset{\vdots\ p_2}{P \triangleleft \{X\}_{-K}}}{P \mid\equiv Q \mid\!\sim X}\ \mathsf{MM2}.$$

- The *nonce verification* or *freshness* rule expresses the check that a message is recent (has been emitted in the current session) and thus that the sender still believes in it. The freshness condition is thus meant against replay attacks:

$$\frac{\overset{\vdots\ a}{P \mid\equiv \sharp(X)} \quad \overset{\vdots\ p}{P \mid\equiv Q \mid\!\sim X}}{P \mid\equiv Q \mid\equiv X}\ \mathsf{NV}.$$

- The *jurisdiction* rule states that if P believes that Q has jurisdiction over X then P trusts Q on the truth of X:

$$\frac{\overset{\vdots\ p}{P \mid\equiv Q \mid\!\Rightarrow X} \quad \overset{\vdots\ a}{P \mid\equiv Q \mid\equiv X}}{P \mid\equiv X}\ \mathsf{J}.$$

- Unsurprisingly, a principal believes a group of statements if and only if it believes each one. We recall that pairs are treated as associative and commutative:

$$\frac{P \mid\equiv X \quad P \mid\equiv Y}{P \mid\equiv (X,Y)}\ \mathsf{BE1} \qquad \frac{\overset{\vdots\ p}{P \mid\equiv (X,Y)}}{P \mid\equiv X}\ \mathsf{BE2}$$

$$\frac{\overset{\vdots\ p}{P \mid\equiv Q \mid\equiv (X,Y)}}{P \mid\equiv Q \mid\equiv X}\ \mathsf{BE3}.$$

Other similar rules may be introduced if necessary, such as

$$\frac{P \mid\equiv Q \mid\equiv X \quad P \mid\equiv Q \mid\equiv Y}{P \mid\equiv Q \mid\equiv (X,Y)}\ \mathsf{BE4}.$$

- Similarly, if a principal said a group of things, it said each of them individually. Note that the converse is not true, since it would imply that the principal said all the things at the same moment:

$$\frac{\overset{\vdots\ p}{P \mid\equiv Q \mid\!\sim (X,Y)}}{P \mid\equiv Q \mid\!\sim X}\ \mathsf{SG}.$$

- If a principal sees a formula, then he also sees its components, provided he knows the necessary keys:

$$\frac{\overset{\vdots\ p}{P \triangleleft (X,Y)}}{P \triangleleft X}\ \mathsf{SP1} \qquad \frac{\overset{\vdots\ p_1}{P \triangleleft \{X\}_K} \quad \overset{\vdots\ p_2}{P \mid\equiv P \overset{K}{\leftrightarrow} Q}}{P \triangleleft X}\ \mathsf{SP2}.$$

Note that the hypothesis is $P \mid\equiv P \overset{K}{\leftrightarrow} Q$, not $P \triangleleft K$, which would seem logical. In fact, this rule could perhaps be replaced by the following pair of rules:

$$\frac{\overset{\vdots\ p}{P \triangleleft \{X\}_K} \quad \overset{\vdots\ a}{P \triangleleft K}}{P \triangleleft X} \qquad \frac{\overset{\vdots\ p}{P \mid\equiv P \overset{K}{\leftrightarrow} Q}}{P \triangleleft K}.$$

The usual rule for public-key cryptosystems is that message encrypted with the public keys are decipherable using the private key:

$$\frac{\overset{\vdots\ p}{P \triangleleft \{X\}_{+K}} \quad \overset{\vdots\ a}{P \mid\equiv \overset{+K}{\mapsto} P}}{P \triangleleft X}\ \mathsf{SP3}.$$

Note that this last rule supposes that if P believes that K is its public key, then it holds the corresponding private key.

The following optional rule expresses the fact that for certain public-key cryptosystems (like RSA [21, 28, 30]), it is possible for anybody with the public key to decipher a message encrypted with the private key:

$$\frac{\overset{\vdots\ p}{P \triangleleft \{X\}_{-K}} \quad \overset{\vdots\ a}{P \mid\equiv \overset{+K}{\mapsto} Q}}{P \triangleleft X}\ \mathsf{SP4}.$$

- If one part of a formula is known to be fresh, then the entire formula must also be fresh:

$$\frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X,Y)}\ \mathsf{FR1} \qquad \frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(\{X\}_K)}\ \mathsf{FR2}.$$

An important point about BAN logic is that it was intended to be a starting point for logics adapted for certain particular uses. A person aiming at applying such techniques to protocols may have to introduce additional constructs and rules to reflect the particularities of the system. The use of automatic decision procedures (see Section 3) may help in this respect to identify the missing rules and assumptions – which sometimes are indeed assumptions about the system that the designer had not noticed.

2.2. The Needham-Schroeder protocol in BAN logic

We shall see here how to formalize and analyze the Needham-Schroeder shared-keys protocol (see Section 1). This protocol is of particular importance since many others, such as Kerberos [22], have been derived from it; furthermore, it has a serious weakness, an undesirable assumption, which can be demonstrated in the logical analysis.

We shall follow the analysis in [10, § 5]. The first step is to convert the protocol description into a sequence of BAN assumptions. Each line of the form $A \to B : F$ induces a formula of the form $B \triangleleft F$. We remove indications that play no role in the logical deductions, such as the names of the principals, and we replace some elements by their semantic meaning: the freshly generated key K_{pq}, meant to be used between P and Q, is idealized as the pair of formulas $P \xleftrightarrow{K_{pq}} Q$ and $\sharp\big(P \xleftrightarrow{K_{pq}} Q\big)$:

$$\begin{aligned}
&2.\ \ S \to P\ :\ \Big\{N_p, P \xleftrightarrow{K_{pq}} Q, \sharp\big(P \xleftrightarrow{K_{pq}} Q\big), \Big\{P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{qs}}\Big\}_{K_{ps}}\\
&3.\ \ P \to Q\ :\ \Big\{P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{qs}}\\
&4.\ \ Q \to P\ :\ \Big\{N_q, P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{pq}}\ \mathit{from}\ Q\\
&5.\ \ P \to Q\ :\ \Big\{N_q, P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{pq}}\ \mathit{from}\ P.
\end{aligned}$$

The first message is omitted, since it does not contribute to the logical properties of the protocol. We should nevertheless not forget that N_p is created just before this first message is sent and thus is assumed to be fresh. The case of the last two messages is more interesting. In the concrete protocol, $N_q - 1$ is used instead of N_q in message 5 so that messages 4 and 5 are different. An intruder cannot replay to P its own message 4. We therefore make this impossibility explicit by using the construction $\Big\{N_q, P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{pq}}\ \mathit{from}\,P$, give in the above explanation of the message-meaning rule.

To start, we give some assumptions:

- The first assumptions state that the principals know how to communicate using shared-key cryptography between the clients and the server:

$$\begin{aligned}
&P \mid\equiv P \xleftrightarrow{K_{ps}} S \qquad Q \mid\equiv Q \xleftrightarrow{K_{qs}} S\\
&S \mid\equiv P \xleftrightarrow{K_{ps}} S \qquad S \mid\equiv Q \xleftrightarrow{K_{qs}} S\\
&S \mid\equiv P \xleftrightarrow{K_{pq}} Q.
\end{aligned}$$

- P and Q trust the server in producing a fresh and correct shared key. They will accept whatever key K that the server will supply; we shall therefore specify these assumptions as *axiom schemes*, where K can be instanced by any value:

$$\begin{aligned}
&\forall K\ P \mid\equiv S \Rrightarrow P \overset{K}{\leftrightarrow} Q, \quad P \mid\equiv S \Rrightarrow \sharp\big(P \overset{K}{\leftrightarrow} Q\big)\\
&\forall K\ Q \mid\equiv S \Rrightarrow P \overset{K}{\leftrightarrow} Q.
\end{aligned}$$

  Since the only value for K that makes sense to reach useful conclusions is K_{pq}, we can replace these axiom schemes by axioms:

$$\begin{aligned}
&P \mid\equiv S \Rrightarrow P \xleftrightarrow{K_{pq}} Q \qquad P \mid\equiv S \Rrightarrow \sharp(P \xleftrightarrow{K_{pq}} Q)\\
&Q \mid\equiv S \Rrightarrow P \xleftrightarrow{K_{pq}} Q.
\end{aligned}$$

  This is also required for our automatic proof technique (Section 3).

- Unsurprisingly, each principal believes in the freshness of what it generates:

$$\begin{aligned}
&P \mid\equiv \sharp(N_p) \qquad Q \mid\equiv \sharp(N_q)\\
&S \mid\equiv \sharp(P \xleftrightarrow{K_{pq}} Q).
\end{aligned}$$

- This last assumption is needed to reach the protocol goals, but is wrong. As pointed out in [10]:

  > [...] the protocol has been criticized for using this assumption, and the authors did not realize they were making it.

  We shall discuss below the unwanted consequences of this assumption:

$$\forall K\ Q \mid\equiv \sharp(P \overset{K}{\leftrightarrow} Q). \tag{1}$$

Let us now see the proofs using BAN logic. First, principal P has to ensure that the key is fresh (Fig. 1). It is then possible to derive $P \mid\equiv P \xleftrightarrow{K_{pq}} Q$ (Fig. 2).

Also, $P \triangleleft \Big\{P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{qs}}$. Since P has seen that part of the message, it can retransmit it to Q. At this point, Q decrypts the message, and we obtain:

$$\frac{Q \mid\equiv Q \xleftrightarrow{K_{qs}} S \quad Q \triangleleft \Big\{P \xleftrightarrow{K_{pq}} Q\Big\}_{K_{qs}}}{Q \mid\equiv S \mid\sim P \xleftrightarrow{K_{pq}} Q}\ \mathsf{MM1}.$$

Let us note that Q has no means to check that this message is fresh except for assumption 1. If we make this assumption, we get

$$\dfrac{\dfrac{Q|\!\equiv S|\!\sim P\xleftrightarrow{K_{pq}}Q \qquad \dfrac{\forall K\ Q|\!\equiv\sharp(P\overset{K}{\leftrightarrow}Q)}{Q|\!\equiv\sharp(P\xleftrightarrow{K_{pq}}Q)}}{Q|\!\equiv S|\!\equiv P\xleftrightarrow{K_{pq}}Q}\,\text{NV} \qquad \dfrac{\forall K\ Q|\!\equiv S\Rightarrow P\overset{K}{\leftrightarrow}Q}{Q|\!\equiv S\Rightarrow P\xleftrightarrow{K_{pq}}Q}}{Q|\!\equiv P\xleftrightarrow{K_{pq}}Q}\,\text{J}.$$

The last two messages are for P and Q to be sure that the other one has indeed received the key and is ready to use it:

$$\dfrac{P|\!\equiv\sharp(P\xleftrightarrow{K_{pq}}Q) \qquad \dfrac{\dfrac{P|\!\equiv P\xleftrightarrow{K_{pq}}Q \quad P\triangleleft\left\{N_q,P\xleftrightarrow{K_{pq}}Q\right\}_{K_{pq}}}{P|\!\equiv Q|\!\sim N_q,P\xleftrightarrow{K_{pq}}Q}\,\text{MM1}}{P|\!\equiv Q|\!\sim P\xleftrightarrow{K_{pq}}Q}\,\text{SG}}{P|\!\equiv Q|\!\equiv P\xleftrightarrow{K_{pq}}Q}\,\text{NV}$$

$$\dfrac{\dfrac{\dfrac{Q|\!\equiv\sharp(N_q)}{Q|\!\equiv\sharp(N_q,P\xleftrightarrow{K_{pq}}Q)}\,\text{FR1} \qquad \dfrac{Q|\!\equiv P\xleftrightarrow{K_{pq}}Q \quad Q\triangleleft\left\{N_q,P\xleftrightarrow{K_{pq}}Q\right\}_{K_{pq}}}{Q|\!\equiv P|\!\sim N_q,P\xleftrightarrow{K_{pq}}Q}\,\text{MM1}}{Q|\!\equiv P|\!\equiv N_q,P\xleftrightarrow{K_{pq}}Q}\,\text{NV}}{Q|\!\equiv P|\!\equiv P\xleftrightarrow{K_{pq}}Q}\,\text{BE3}$$

In other words, each principal P or Q trusts the other one in believing they share a secret key K_{pq}.

Let us now discuss the weakness of the protocol: assumption 1 $\left(\forall K\ Q|\!\equiv\sharp(P\overset{K}{\leftrightarrow}Q)\right)$. It means that Q will accept a proposal for a key K_{pq} without being able to check whether this key is appropriate for this session. In fact, let us suppose that an intruder I has listened to the network and recorded a session involving P and Q. It therefore has recorded a valid message $\left\{K_{pq},P\right\}_{K_{qs}}$. Let us additionally assume that the intruder has managed to get hold of K_{pq}. Now the intruder impersonates P to initiate a run of the protocol with Q using that recorded information:

$$\begin{aligned}
3.\ \ I\rightarrow Q &: \left\{K_{pq},P\right\}_{K_{qs}}\\
4.\ \ Q\rightarrow I &: \left\{N_q\right\}_{K_{pq}}\\
5.\ \ I\rightarrow Q &: \left\{N_q-1\right\}_{K_{pq}}.
\end{aligned}$$

Now Q believes it can communicate with P using K_{pq}. Q will in good faith start talking with the intruder, believing the intruder is P. In other words, if one session key has been compromised, all subsequent sessions can be compromised as well. This contradicts one of the very motivations for the use of session keys which is "to limit exposure, with respect to both time period and quantity of data, in the event of (session) key compromise" [21, §12.2.2]. As [10] points out:

> Denning and Sacco pointed out that compromise of a session key can have very bad results: an intruder has unlimited time to find an old session key and to reuse it as though it were fresh (1981). Bauer, Berson, and Feiertag pointed out that there are even more drastic consequences if [P]'s private key is compromised: an intruder can use [P]'s key to obtain session keys to talk to many other principals, and can continue to use these session keys even after [P]'s key has been changed (1983). It is comforting that the logical analysis makes explicit the assumption.

BAN logic has thus been successful in identifying an unwanted assumption of a protocol on the freshness of a message, indicating the possibility of a replay attack.

2.3. A simple example in GNY logic

BAN logic was much criticized, on the one hand for being some kind of dubious idealization of the already idealized Dolev-Yao model, on the other hand for making unwanted assumptions. We have already seen the uneasy treatment that BAN makes of the situation where a principal P is sent a message $\{M\}_{K_{pq}}$, where K_{pq} is a shared key for P and Q: P can believe that this message originated from Q only if P is sure that this message is not a replay of one of its own messages. GNY logic is a BAN-like logic that solves this issue as well as others [17]:

> Our new approach seems to offer important advantages over the BAN approach. It does not require several universal assumptions which the BAN work does. For example, it does not assume that redundancy is always present in encrypted messages incorporating instead a new notion of recognizability which captures a recipient's expectation of the contents of messages he receives. Also, it does not assume that a principal can always determine whether a message was not once originated by himself.

GNY logic also separates what a principal says, what it believes and what it possesses. Other logics have been proposed to alleviate some other weaknesses of BAN logic [33, 34]. We shall restrict ourselves here to a cursory glance at GNY logic [17].

We shall consider a very simple (and admittedly silly) protocol:

$$\begin{aligned}
1.\ \ A\rightarrow B &: N_a\\
2.\ \ B\rightarrow A &: \{N_a\}_{-K_b}\\
3.\ \ A\rightarrow B &: \left\{K_{ab}\right\}_{+K_b}.
\end{aligned}$$

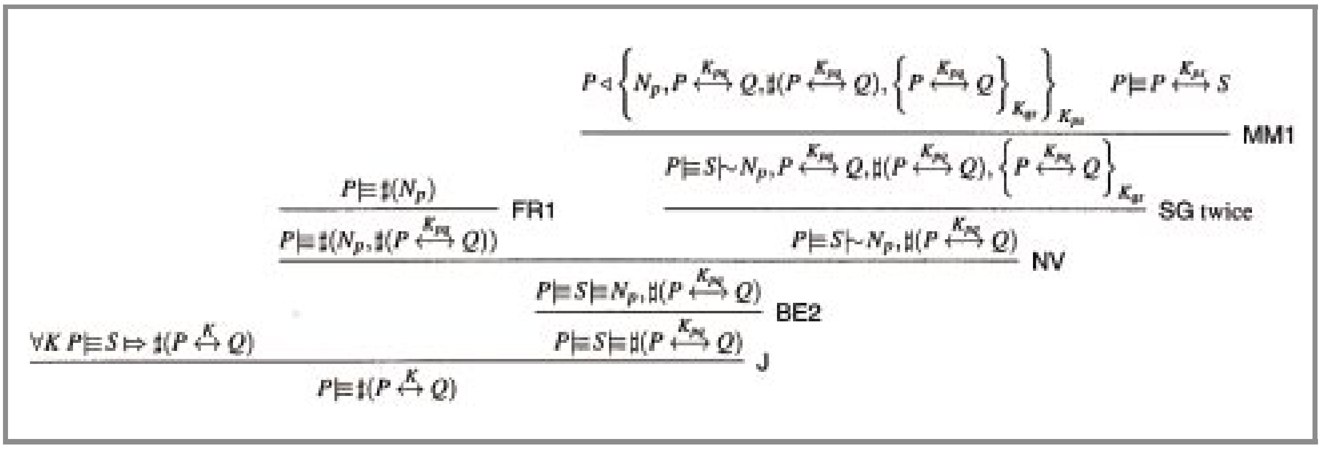

Fig. 1. Deriving freshness in BAN logic.

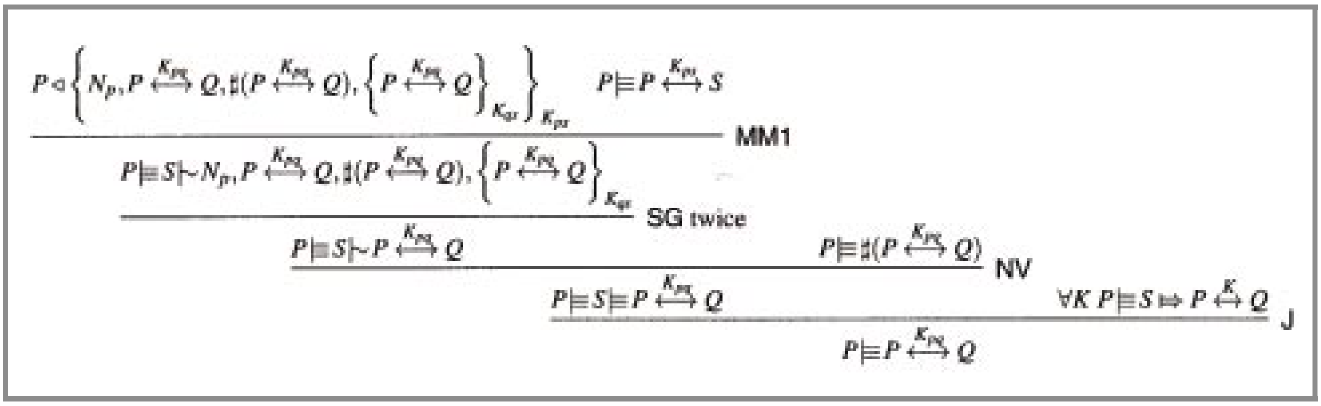

Fig. 2. Deriving that a key is shared in BAN logic.

This is to be understood as: in step 1, A sends a newly generated number N_a to B; B answers with the encryption of N_a by its private key $-K_b$; A answers with the encryption of a newly generated session key K_{ab} with B's public key $+K_b$.

To illustrate how belief-logic deductions work, we first show the idealized version of the protocol in GNY:

1. $B \triangleleft N_a$
2. $A \triangleleft \star \{N_a\}_{-K_b}$
3. $B \triangleleft \star \left(\{K_{ab}\}_{+K_b} \rightsquigarrow A \overset{K_{ab}}{\longleftrightarrow} B \right)$.

The star means that the following term was not originated by the party who receives it. The statement after the wavy arrow in 3 is an annotation meaning that K_{ab} is intended to be a shared secret key for use between A and B.

We also need some assumptions, written as follows in GNY:

(a) $A \ni +K_b$ (A possesses $+K_b$)

(b) $A \models \overset{+B}{\mapsto} +K_b$ (A believes that $+K_b$ is B's public key)

(c) $A \models \phi(N_a)$ (A believes N_a to be recognizable; that is, if A sees a message field that is supposed to be N_a, A can check whether it is or not)

(d) $A \models \sharp(N_a)$ (A believes N_a to be fresh; that is, to have been used for the first time in this run of the protocol).

See [17] for a complete list of the GNY inference rules and their designations. Using those rules, we can derive the conclusion $A \models B \ni N_a$ as in Fig. 3.

This means that from the fact, coming from protocol step 2, that A sees the message consisting of the encryption of N_a by the private key for the public/private couple of keys K_b, from assumption a (A possesses the corresponding public key), from assumption b (B uses the private key of the couple K_b) and from assumption c (A believes that it can recognize N_a), we deduce using rule I4 that A is entitled to believe that B once said N_a. Then, using assumption d, which is that N_a is fresh (has never been used in another session before), we deduce that A is entitled to believe that B possesses N_a.

The final goal of the protocol might be to cause B to believe that A believes that K_{ab} is the shared key ($A \models A \overset{K_{ab}}{\longleftrightarrow} B$). However, message 3 could have been forged by any intruder possessing $+K_b$, which is realistic since it is a public key,

$$\dfrac{\dfrac{A \triangleleft \star\{N_a\}_{-K_b} \quad A \ni +K_b \quad A \mid\equiv \overset{+K_b}{\mapsto} +K_b \quad A \mid\equiv \phi(N_a)}{A \mid\equiv B \mid\sim N_a}\,\mathsf{I4} \qquad A \mid\equiv \sharp(N_a)}{A \mid\equiv B \ni N_a}\,\mathsf{I6}$$

Fig. 3. A derivation in GNY logic.

replacing K_{ab} by any key of his choice. The logic (correctly) fails to conclude that the protocol accomplishes this goal: this goal has no derivation in GNY logic from the above set of hypotheses.

3. Decidability

A little known fact about the modal logics of belief (at least BAN and GNY) is that they are *decidable* [23]. That is, there exists an algorithm that, given a finite set of hypotheses H_1 to H_n and a purported conclusion C, answers whether or not C follows from $H_1, \dots, H_n$. We shall see in this section the difficulties of establishing this property and a practical algorithm.

3.1. Position of the problem

Let us first remark that not all logics are decidable. For instance, set theory, the basis of usual mathematics, is *undecidable* [12]: that is, there exists no algorithm that takes as input a mathematical proposition and answers whether it is true or false. Furthermore, the analysis of cryptographic protocols in the Dolev-Yao model, given some very reasonable hypotheses, is also undecidable if an unbounded number of sessions is allowed, even with messages of bounded depth [14] [3].

There are two traditional methods to test whether a formula t admits a derivation from a set of hypotheses Γ in a rule system $\vdash$ (which we note by $\Gamma \vdash t$):

- **Forward chaining**, that is starting from the hypotheses Γ, apply all the possible deduction rules to deduce new formulas, then start again with the union of the hypotheses and the new formulas, until the formula t is discovered.

- **Backward chaining**, that is, starting from the purported conclusion, find all the rules and all the instantiations of the variables in them that yield that conclusion, then try recursively to prove the hypotheses of each of these rules with each of the instantiations; this requires backtracking.

There are two problems with the rule systems like BAN or GNY:

- Both forward chaining and backward chaining may fail to terminate.

- There are rules that are unsuitable for forward-chaining and rules that are unsuitable for forward-chaining. For instance,

$$\frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X,Y)}\ \mathsf{FR1}$$

  has a conclusion in which there are variables that are not found in the hypotheses. It is therefore impossible to apply forward chaining, except by introducing variables representing unknown formulas. Similarly, rule BE2 is not suitable for backward-chaining.

If we straightforwardly (and naively) implement the rules of BAN or GNY logic in a general-purpose automatic theorem prover, as it has been done [31], the prover is likely to search an infinite space of possible proofs, which means that the system does not terminate when the conclusion is not provable. Furthermore, even in cases of termination, the computation time might be prohibitive, because the search procedure explores many useless avenues.

The approach taken by Kindred and Wing [19] and generalized by ourselves [23] is a refinement on a combination of forward and backward chaining. We shall expose here briefly our method. This method analyses the GNY logic [17], but is generic enough to be applied to most similar logics. It is based on a careful application of forward-chaining.

3.2. Composition and decomposition rules

Let us take a look at BAN logic[4]. We consider a partition of the rules between these two classes:[5]

- *Decomposition rules*, in which all the variables of the conclusion are found in the premises (these rules are suitable for forward-chaining); for these rules, we distinguish the *principal premises* (which can be one or more) and the optional *auxiliary premises*; the

[3] See also Comon and Shmatikov's paper in this volume.

[4] We do similar work for a variant of GNY logic equivalent to GNY logic in [23].

[5] This partition of the set of rules into two classes is very similar to that of [19], where they are called respectively *growing* and *shrinking rules*. This is also similar to the introduction and elimination rules of natural deduction; see [15, p. 75] and [27 § II.1]. Our theorem on normal derivations is thus similar to the normalization theorem of natural deduction [27, § IV.1] or Gentzen's *Hauptsatz* [15, p. 105].

variables in the auxiliary premises are a subset of these in the principal premises; those rules in BAN are the ones listed above as

$$\frac{\overset{\vdots\, p_1}{\mathscr{H}_1} \quad \cdots \quad \overset{\vdots\, p_n}{\mathscr{H}_n} \quad \overset{\vdots\, a_1}{\mathscr{H}_{n+1}} \quad \cdots \quad \overset{\vdots\, a_m}{\mathscr{H}_{n+m}}}{\mathscr{C}}$$

- *Composition rules*, in which all the variables of the premises are found in the conclusion (these rules are suitable for backward-chaining).

The intuition is that composition rules introduce constructors, like a pair, and decomposition rules break these constructors, as in taking the first projection of a pair.

GNY and similar logics fulfill the *normal derivation criterion*: if there exists a derivation of $\Gamma \vdash t$ then there must also exist a *normal derivation* of $\Gamma \vdash t$. A derivation Δ of a conclusion $\Gamma \vdash t$ is said to be *normal* if there is no composition rule to be used as the root rule of the sub-derivation for a principal premise of a decomposition rule: for any decomposition rule d used in Δ:

$$\frac{\frac{\vdots}{\mathscr{P}_1}\,r_1 \quad \cdots \quad \frac{\vdots}{\mathscr{P}_n}\,r_n \quad \overset{\vdots}{\mathscr{A}_1} \quad \cdots \quad \overset{\vdots}{\mathscr{A}_m}}{C}\,d$$

where $\mathscr{P}_i$ are the principal hypotheses and $\mathscr{A}_j$ the auxiliary hypotheses, none of the rules $r_1, \ldots, r_n$ is a composition rule. In the opposite case, we say that there is a *detour* at r.

Informally, that means that it must be possible to derive anything that is derivable without having to compose something and decompose it afterwards;[6] for instance, in

$$\frac{\dfrac{\overset{\vdots\,\alpha}{P \models X} \quad \overset{\vdots\,\beta}{P \models Y}}{P \models (X,Y)}\,\text{BE1}}{P \models X}\,\text{BE2}$$

we compose a pair just to decompose it afterward, and we could have reached the same conclusion directly using only the α branch of the proof:

$$\overset{\vdots\,\alpha}{P \models X}\,.$$

3.3. Magic-set transformation

As we pointed out earlier, it is possible to implement straightforwardly BAN or GNY logic into forward-chaining or backward-chaining (Prolog-like) theorem provers; this is nevertheless clumsy since:

- It is necessary to allow non-ground formulas, that is, formulas containing variables representing unknown sub-formulas. This means that the prover needs apply unification and not just pattern-matching when applying rules. This makes the implementation far more complex.

- Many unnecessary conclusions or intermediary goals are generated. In the case of BAN or GNY logic, this leads to non-termination when the purported conclusion is not reachable from the hypotheses.

Similar problems were encountered in the field of logic programming and queries in logical databases and led to the introduction of the *magic-set* transformation [7]. This transformation turns backward-chaining rules into forward-chaining ones, by generating only "relevant" sub-goals.[7] These sub-goals constitute the so-called "magic sets" associated with the predicates. The basic idea is that we should only begin investigating sub-goals of a conclusion only if some sub-goals have already been proved and some variables instanciated (thus the vocabulary of *Sideways Information Passing Strategy* or *SIPS*).

In our case, the partitions into composition and decomposition rules, and in that latter class, between principal and auxiliary premises gives us the SIPS. Furthermore, we choose our partition so that all the forward-chaining rules generated by the transformation make some measure decrease, thus ensuring termination. We then define another logic, introducing a special symbol, *goal*. $\boxdot X$ means that "we would like to compose X". Other authors call this symbol "magic".

Our transformation turns the composition rule

$$\frac{\mathscr{H}_1 \cdots \mathscr{H}_n}{\mathscr{C}}$$

into a pair

$$\frac{\boxdot\mathscr{C} \quad \mathscr{H}_1 \cdots \mathscr{H}_n}{\mathscr{C}} \qquad \frac{\boxdot\mathscr{C}}{\boxdot\mathscr{H}_1 \cdots \boxdot\mathscr{H}_n}$$

and adds to the decomposition rule

$$\frac{\mathscr{P}_1 \cdots \mathscr{P}_n \quad \mathscr{A}_1 \cdots \mathscr{A}_m}{\mathscr{C}},$$

where the one or more $\mathscr{P}_i$ are principal premises and the zero or more $\mathscr{A}_i$ are auxiliary premises, the triggering rule

$$\frac{\mathscr{P}_1 \cdots \mathscr{P}_m}{\boxdot\mathscr{A}_1 \cdots \boxdot\mathscr{A}_n}.$$

It can be proved [23] that

Theorem 1: For any set of hypotheses $\mathscr{H}$ and purported conclusion $\mathscr{C}$ for the $\vdash$ proof system,

$$\mathscr{H} \vdash \mathscr{C} \iff \mathscr{H}, \boxdot\mathscr{C} \vdash' \mathscr{C}.$$

[6] In terms of natural deduction and similar systems, this is often called the *inversion principle* [27, ch. II].

[7] As it has been pointed out [7, § II], "relevant" means "[that might] be essential to the establishment of a fact that is in the answer". "Relevant" goals may actually be irrelevant to the query; the only insurance that we have is that for any provable fact there will be one proof of that fact for which we shall obtain all sub-goals as "relevant".

3.4. The decision procedure

The interest of turning the original problem on $\vdash$ into a problem on $\vdash'$ is twofold:

1. All rules in $\vdash'$ are suitable for forward-chaining.
2. For any finite set of hypotheses $\mathscr{H}$, the length of the derivations of the $\vdash'$-proofs starting from $\mathscr{H}$ is bounded. This condition is proved by giving a *weight function* assigning an integer weight $|F|$ to each formula F so that the weight of the conclusion of a $\vdash'$-rule is strictly less than the maximal weight of the premises[8] (Table 1).

To some extent, the existence of the weight function is intuitive: some rules are evidently composition rules (building a pair, for instance) or decomposition rules (splitting a pair), and the intuition is that the conclusion of a decomposition rule is "smaller" than the premises, and that the conclusion of a composition rule is "bigger" than the premises. The additional condition that the auxiliary premises are smaller than the principal ones is also intuitive, since the auxiliary premises often represent keys and the principal premises represent encrypted data containing the key as a sub-term. The rules resulting from the magic-set transformation then have conclusions that are "smaller" than their premises.

Table 1
The weight function for BAN logic

F	$\|F\|$
K	1
P	1
N_p	1
$P \mid\equiv X$	$1 + \|X\|$
$P \triangleleft X$	$2 + \|X\|$
$P \mid\sim X$	$3 + \|X\|$
$P \Rightarrow X$	$3 + \|X\|$
$\sharp(X)$	$1 + \|X\|$
$P \overset{K}{\leftrightarrow} Q$	$2 + \|K\|$
$\overset{+K}{\mapsto} P$	$2 + \|K\|$
$\{X\}_K$	$3 + \|X\| + \|K\|$
(X, Y)	$1 + \|X\| + \|Y\|$

For any finite set $\mathscr{H}$ of hypotheses, the set of conclusions that can be derived from $\mathscr{H}$ is finite and can be enumerated by exhaustively applying the rules of $\vdash'$; this is often referred to as the *saturation* of the hypotheses by the forward-chaining system $\vdash'$. The decision procedure for $\vdash'$ is thus simple: to test whether $A \vdash' B$, it suffices to saturate A by $\vdash'$ and test whether B belongs to the set. Let us note that although $\mathscr{W}$ is used to prove the termination of the saturation process, that process does not need to compute $\mathscr{W}$.

Using Theorem 1, we obtain a decision procedure for $\vdash$ (which can be BAN logic or a modified version of GNY logic [23]).

Minimal care must be taken when implementing the saturation procedure, especially for more complex logics such as GNY. Naive implementations may lead to prohibitive costs. For instance, trying all possible rules in the fashion that to try a n-ary rule you match it against all the n-tuples of already derived formulas, until it ends, leads to prohibitive costs (in the case of GNY, $n = 6$, which makes the number of matchings grow in D^7, where D is the number of derivable formulas).

A first optimization we tried was based on the fact that one need not test all possible n-tuples, but only ones containing at least one "new" formula; that is, a formula made during the last application of the rules. This is not sufficient, since it reduces only to D^6; experimentally, this is far too slow.

Our implementation is based on the fact that to instantiate all the variables in a rule, you need not consider all the hypotheses; especially, in the modified versions of the composition rules, only one "goal" hypothesis suffices; in the decomposition and trigger rules, only the principal premises are needed. Expensive exhaustive searches for the fully instantiated hypotheses are replaced by a much faster binary search. On problems taken from the protocol literature, this implementation performed within seconds.[9] Implementations based on efficient general-purpose forward chaining systems are likely to perform even better.

Our goal is not only to "prove" protocols in the logic, but also to identify undesirable assumptions. Our analyzer can also help in that regard, since it generates a list of possibly desirable assumptions (the formulas F so that $\boxdot F$ is in the saturation of the problem by $\vdash'$, while F is not). Experimentally, the list given by the analyzer tends to contain the missing assumptions, but also many ludicrous ones. Heuristics may help to produce meaningful output to the protocol designer.

4. Semantics

We have so far defined a system of rules. But are we sure that they are the right rules? Are we sure we are not going to deduce something wrong because of a loophole in the system? It would be much better if we had a way of representing the concepts that are embodied in the formulas and check whether the deductive relationships expressed by the rules are actually true. For this, we must define the *semantics* of formulas in terms of *models*.

We shall remind the reader briefly of the semantic aspects of axiomatic methods. Let us take a simple example. It is possible to write proofs of facts in planar geometry using a few structural rules (*modus ponens* and other rules for basic logical connectors) as well as a few axioms on geometrical properties. This is the usual geometry that children learn at school. On the other hand, it is possible to

[8] The problem is slightly more complex for GNY logic because of its *jurisdiction rule* [23].

[9] The implementation is freely available from the author.

build a theory of geometry based on set theory, the integer, the rationals, the real field and Euclidean spaces. What is the relationship between those two theories? Every object (point, line …) of usual planar geometry can be represented by an object built from set theory. The same holds for the relationships between those objects (parallelism, intersections …) and even whole statements. These representations constitute a *model*. We say that a model $\mathscr{M}$ represents a formula F (noted $\mathscr{M} \models F$) if and only if the representation of the formula F is true in the model $\mathscr{M}$. A statement is said to be *valid* if it is true in all models. There are then two problems to consider:

- Is the system of rules that we consider *sound*? That is, are all deducible statements true in all models?
- Is the system of rules complete? That is, is there a proof for every valid statement that can be written in the system?

It is interesting to note that indeed it is possible to give a sound and complete axiomatic system for planar geometry [37] and quite a few other interesting theories.

Early attempts at giving semantics for logics of belief for cryptographic protocols gave only somehow "trivial" semantics: they basically said that what a principal believes is what it had come to believe following the rules. Such a semantics does not shed any light on what "belief" means in the context of cryptographic protocols. It seems desirable to have logics of belief proved to be sound with respect to a non-trivial semantics for beliefs, which will involve a notion of *possible worlds* [18]. Improved belief logics, proved to be sound with respect to possible world semantics, were therefore proposed [4, 34]. Later, semantics based on *strand spaces* were also proposed [35].

5. Conclusions

BAN, and similar logics, are useful to get an idea of the assumptions underlying the design of a cryptographic protocol. Their handling can be (partially) automated. While they do not provide the same sort of assurance as analyses in the Dolev-Yao model [13] or the spi-calculus and its variants [1–3], they can point mistakes in the design of protocols, including misplaced trust or failure to prevent replay attacks.

Acknowledgements

We thank J. Goubault-Larrecq, Jon Millen, and Luca Vigano for their many suggested improvements to the paper.

References

[1] M. Abadi and A. D. Gordon, "A calculus for cryptographic protocols: the spi calculus", in *ACM Conf. Comput. Commun. Secur.*, 1997, pp. 36–47.

[2] M. Abadi and A. D. Gordon, "Reasoning about cryptographic protocols in the spi calculus", in *CONCUR'97, Concurrency Theory, 8th International Conference*, A. Mazurkiewicz and J. Winkowski, Eds., *Lecture Notes in Computer Science.* Springer, 1997, vol. 1243, pp. 59–73.

[3] M. Abadi and A. D. Gordon, "A calculus for cryptographic protocols: the spi calculus". Res. Rep. 149, Compaq Systems Research Center, Palo Alto, CA, USA, Jan. 1998.

[4] M. Abadi and M. R. Tuttle, "A semantic for a logic of authentication", in *10th Annual ACM Symposium on Principles of Distributed Computing,* L. Logrippo, Ed. ACM Press, 1991, pp. 201–216.

[5] R. Amadio and D. Lugiez, "On the reachability problem in cryptographic protocols". Res. Rep. 3915, INRIA, 2000.

[6] R. Amadio and D. Lugiez, "On the reachability problem in cryptographic protocols", in *CONCUR'00, Lecture Notes in Computer Science.* Springer, 2000, vol. 1877.

[7] F. Bancilhon, D. Maier, Y. Sagiv, and J. D. Ullman, "Magic sets and other strange ways to implement logic programs (extended abstract)", in *International Conference on Management of Data and Symposium on Principles of Database Systems.* ACM, 1986.

[8] D. A. Basin, "Lazy infinite-state analysis of security protocols", in *Secure Networking – CQRE (Secure) '99, International Exhibition and Congress*, R. Baumgart, Ed., *Lecture Notes in Computer Science*. Springer, 1999, vol. 1740, pp. 30–42.

[9] P. Bieber, "A logic of communication in hostile environment", in *Comput. Secur. Found. Worksh. (III)*, 1990, pp. 14–22.

[10] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication". Tech. Rep. 39, Digital Equipment Corporation, Systems Research Centre, Febr. 1989.

[11] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication", *ACM Trans. Comput. Syst.*, vol. 8, no. 1, pp. 18–36, 1990.

[12] A. Church, "An unsolvable problem of elementary number theory", *Am. J. Math.*, vol. 58, no. 2, pp. 345–643, 1936.

[13] D. Dolev and A. C. Yao, "On the security of public key protocols", *IEEE Trans. Inform. Theory*, vol. IT-29, no. 12, pp. 198–208, 1983.

[14] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov, "Undecidability of bounded security protocols", in *Proc. Worksh. Form. Meth. Secur. Protoc. (FMSP)*, N. Heintze and E. Clarke, Eds., 1999.

[15] J. Y. Girard, *Proof and Types*. Cambridge University Press, 1990. (Translated and with appendices by Paul Taylor, Yves Lafont).

[16] L. Gong, "Cryptographic protocols for distributed systems". Ph.D. thesis, University of Cambridge, Cambridge, England, April 1990.

[17] L. Gong, R. Needham, and R. Yahalom, "Reasoning about belief in cryptographic protocols", in *IEEE Symposium on Research in Security and Privacy*. IEEE Computer Society Press, 1990, pp. 234–248.

[18] J. Hintikka, *Knowledge and Belief: An Introduction to the Logic of Two Notions*. Cornell University Press, 1962.

[19] D. Kindred and J. M. Wing, "Fast, automatic checking of security protocols", in *Second USENIX Workshop on Electronic Commerce*. USENIX, 1996, pp. 41–52.

[20] C. Meadows, "The NRL protocol analyzer: an overview", *J. Log. Program.*, 1995 (to appear).

[21] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.

[22] S. P. Miller, B. C. Neuman, J. I. Schiller, and J. H. Saltzer, "Kerberos authentication and authorization system", in *Project Athena Technical Plan*. MIT, 1987, ch. E.2.1.

[23] D. Monniaux, "Decision procedures for the analysis of cryptographic protocols by logics of belief", in *12th Computer Security Foundations Workshop*. IEEE, 1999.

[24] R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers", *Commun. ACM*, vol. 21, no. 12, 1978.

[25] D. Nessett, "A critique of the Burrows, Abadi and Needham logic", *ACM Oper. Syst. Rev.*, vol. 24, no. 2, pp. 35–38, 1990.

[26] L. C. Paulson, "Proving properities of security protocols by induction", in *10th Computer Security Foundations Workshop*. IEEE Computer Society Press, 1997, pp. 70–83.

[27] D. Prawitz, *Natural Deduction: a Proof-Theoretical Study*. Almqvist & Wiksell, Stockholm, 1965.

[28] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems", in *SIMMONS: Secure Communications and Asymmetric Cryptosystems*, 1982.

[29] M. Rusinowitch and M. Turuani, "Protocol insecurity with finite number of sessions is NP-complete", in *14th IEEE Comput. Secur. Found. Worksh.*, Cape Breton, Nova Scotia, Canada, June 2001.

[30] B. Schneier, *Applied Cryptography*. 2 ed. Wiley, 1996.

[31] J. Schumann, "Automatic verification of cryptographic protocols with SETHEO", in *Proceedings of the 14th International Conference on Automated Deduction*, W. McCune, Ed., *Lecture Notes in Artificial Intelligence*. Berlin: Springer, 1997, vol. 1249, pp. 87–100.

[32] D. Song, "Athena: a new efficient automatic checker for security protocol analysis", in *12th Computer Security Foundations Workshop*. IEEE, 1999.

[33] P. Syverson, "Adding time to a logic of authentication", in *1st ACM Conf. Comput. Commun. Secur.*, 1993, pp. 97–101.

[34] P. Syverson and P. C. van Oorschot, "On unifying some cryptographic protocol logics", in *1994 IEEE Comput. Soc. Symp. Res. Secur. Priv.*, May 1994, pp. 14–28.

[35] P. Syverson, "Towards a strand semantics for authentication logics", in *Electronic Notes in Theoretical Computer Science*. 1999, vol. 20.

[36] P. Syverson and I. Cervesato, "The logic of authentication protocols", in *FOSAD'00*, R. Focardi and R. Gorrieri, Eds. Springer, 2001, vol. 2171.

[37] A. Tarski, "What is elementary geometry?", in *The Axiomatic Method, with Special Reference to Geometry and Physics*, P. Suppes and A. Tarski, Eds., *Studies in Logic and the Foundations of Mathematics*. North-Holand, 1959, pp. 16–29.

David Monniaux is a researcher at the Department of Computer Science at Ecole Normale Supérieure in Paris, France. His research interests include formal methods, program semantics and abstract interpretation, and the static analysis of cryptographic protocols, critical software, and probabilistic programs. He received his Ph.D. degree in computer science from Université Paris-Dauphine in 2002 and is an alumnus of Ecole Normale Supérieure de Lyon.

e-mail: David.Monniaux@ens.fr
École Normale Supérieure
Département d'Informatique
45, rue d'Ulm
75230 Paris Cedex 05, France