Towards Formal Analysis of Security Protocols*

Wenbo Mao

Communications Research Lab
Department of Electrical Engineering
University of Manchester
Manchester M13 9PL
England

Tel: +44 61 275 4506
wenbo@uk.ac.man.ee.comms

Colin Boyd

Communications Research Lab
Department of Electrical Engineering
University of Manchester
Manchester M13 9PL
England

Tel: +44 61 275 4562
colin@uk.ac.man.ee.comms

Abstract

The pioneering and well-known work of Burrows, Abadi and Needham (the BAN logic) which dominates the area of security protocol analysis is shown to take an approach which is not fully formal and which consequently permits approval of dangerous protocols. Measures to make the BAN logic formal are then proposed. The formalisation is found to be desirable not only for its potential in providing rigorous analysis of security protocols, but also for its readiness for supporting a computer-aided fashion of analysis.

1 Introduction

A security protocol such as one for distributing cryptographic keys is essentially a few lines of a specification of a program. Its analysis can therefore be considered as analogous to the correctness verification of such a program. However, unlike the case of running a computer program, where the user naturally bears an intention to follow the instruction so to avoid potential bugs, the main objective of a dishonest user during a run of a security protocol is to exploit its bugs and through the abuse to obtain or alter vital information. The damage caused can be disastrous. Hence, debugging of a security protocol must be so thorough that its abuse can achieve nothing.

Such thorough debugging is possible only through formal techniques. Hence, analysis of protocols with formal logical approaches have been widely taken in this area of study. A suitable category of logical approaches in this area are identified as the so-called *logics of belief* e.g. [2, 9, 11, 4, 10]. They reason about beliefs of principals (people, computers and so on) on the security properties of the communication channels. In the past a few years the success of these approaches was best demonstrated by the leading work of Burrows, Abadi and Needham, the BAN logic. In their seminal paper [2] the logic was shown to be good at revealing various subtle security flaws and drawbacks in several authentication protocols. In addition to these, a number of others have also applied BAN logic to successfully verify other security protocols (see e.g. the CCITT X.509 strong two-way authentication protocol in [8] and a multi-party session protocol in [4]). Furthermore, extensions to the logic were reported in [9]. The last two papers also made it clear that a logic of belief can potentially support a computational, or a computer-aided, reasoning method.

However successful, critiques of BAN logic on various features have been published. First, Nessett criticised BAN logic about its claimed goals of authentication [12]; he constructed a simple example to demonstrate the logic's failure to discover flaws which violate security in a basic sense. Secondly, Snekkenes examined the logic's limitation of providing only partial correctness proofs [14], a worry that was actually noticed as early as October 1989 [6]; namely, when the logic finds a bug in a protocol, everyone believes that it is a bug; when the logic finds a proof of correctness, people seem to have trouble believing that it is a proof. Thirdly, Syverson explained a problem of informality in the logic's operational semantics [15]; he

*This work is funded by the UK Science and Engineering Research Council under research grant GR/G19787.

also described confusions about the goals of the logic. Finally, in [11], Moser argued that unlike knowledge, a belief is refutable. However, this is not the case in BAN logic, where any belief once established is irrefutable, even though it can be utterly false.

It is our recognition that known subsequent proposals of logics of belief (e.g. those reported in [9, 11, 10]) still leave the above problems more or less unsolved. Taking their scenarios of belief establishment and/or development for instance, every of these subsequent logics has recourse to some non-neutral statements obtained from an insecure channel. As will be made clear in this paper, such a treatment forms a major misleading interpretation over the semantics of peer-entity authentication.

To tackle these problems forms the motivation of the research reported in this paper. Reasons why BAN logic gives rise to these problems will be discussed and measures of their rectification attempted. The rectification results in a logic which essentially adopts the notational framework of BAN logic but is equipped with a new scenario of belief. For simplicity, the logic presented in this paper only deals with key-distribution protocols where a key-distribution server is needed.

The remainder of the paper is organised as follows. Section 2 is a brief introduction to BAN logic, whose deficiencies are examined in Section 3. Rectification measures for these problems are proposed in Section 4. The revised logic is then applied in Section 5 to demonstrate protocol analysis examples. Finally, Section 6 gives our conclusion.

2 BAN Logic

BAN logic can be viewed as a predicate logic constructed on several sorts of objects: principals, encryption keys, messages and formulas (also called statements). Typically, capital letters such as $A, B...$ are used to denote specific principals; K_{ab} denotes the specific shared key; K_a and K_b specific public keys, and K_a^{-1} and K_b^{-1} denote the corresponding secret keys; a piece of information which is cryptographically enciphered by using a key K is denoted $\{X\}_K$; without possession of the key K one is unable to read the message X.

Predicate constructs are used to interpret organised objects into logical statements with truth values. They include a number of important and frequently used constructs as listed below.

$P|\equiv X$: P believes X; the principal P may act as though the statement X is true.

$P|\sim X$: P once said X; the principal P at some time said the statement X.

$P \lhd X$: P sees X; someone has sent the statement X, P can read and repeat X.

$P \Rightarrow X$: P has jurisdiction over X; the principal P is an authority on the truth of X and should be trusted on this matter.

$\sharp(X)$: X is fresh; it has not been sent at any time before the current protocol run.

$P \overset{K}{\leftrightarrow} Q$: P and Q may use the shared encipher 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.

A set of inference rules (called "logical postulates" in [2]) are used as a means to reason about the above predicate constructs. Important and frequently used ones are listed below.

The message-meaning rule. The following rule formalises one of the main semantical principles of BAN logic; namely, if you believe that you and Joe know a key, then you ought to believe that anything you receive enciphered with the key comes originally from Joe:

$$\frac{P|\equiv Q \overset{K}{\leftrightarrow} P,\ P \lhd \{X\}_K}{P|\equiv Q|\sim X}$$

where symbol "," is the boolean conjunctive operator with the following operational semantics: statement S_1, S_2 is true if and only if both S_1 *and* S_2 are true.

The nonce-verification rule. The following rule expresses the check that a statement is recent, and hence that the sender still believes in it:

$$\frac{P|\equiv \sharp(X),\ P|\equiv Q|\sim X}{P|\equiv Q|\equiv X}$$

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

$$\frac{P|\equiv Q \Rightarrow X,\ P|\equiv Q|\equiv X}{P|\equiv X}$$

The freshness rule. The following rule is important in reflecting the notion of timeliness exploited as the cardinal principle of authentication:

$$\frac{P \models \sharp(X)}{P \models \sharp(X,Y)}$$ [1]

There are a few other inference rules. We shall not list them one by one.

A protocol analysis with BAN logic consists of the following steps. First, a protocol is transformed into a so-called *idealised* form; the transformation involves not only protocol syntax changes, but also semantic interpretations. Secondly, logical formulas about the (idealised) protocol are generated and reasoned about by applying the inference rules. The reasoning manipulation starts from a set of formulas called *initial assumptions*; guided by the idealised protocol specification, it aims at reaching another set of formulas called *conclusions*. For instance, the objective of analysing a key-distribution protocol is to see if it is possible to establish some desirable formulas that describe beliefs of the goodness of the distributed key, namely, formulas like $A \models A \overset{K}{\leftrightarrow} B$.

3 The Defects of BAN Logic

We first note that the BAN logic has difficulty to debug certain protocols which have serious security defects. One such protocol is an optimised version of Otway-Rees protocol [13] which was suggested by the authors of BAN logic in [2]. The protocol is given below.

1 $A \rightarrow B : \ M, A, B, \ \{N_a, M, A, B\}_{K_{as}}$
2 $B \rightarrow S : \ M, A, B, \ \{N_a, M, A, B\}_{K_{as}}, \ N_b, \{M, A, B\}_{K_{bs}}$
3 $S \rightarrow B : \ M, \{N_a, K_{ab}\}_{K_{as}}, \{N_b, K_{ab}\}_{K_{bs}}$
4 $B \rightarrow A : \ M, \{N_a, K_{ab}\}_{K_{as}}$

Indeed, as pointed out by the authors of BAN logic, desirable formulas $A \models A \overset{K_{ab}}{\leftrightarrow} B$ and $B \models A \overset{K_{ab}}{\leftrightarrow} B$ are derivable when using their logic to analyse this protocol. However, an attack demonstrates that the above two beliefs are groundless for there is a disastrous error in the protocol. In the attack, an attacker C masquerades as A in the protocol. It is assumed that C has possession of a message fragment $\{M', C, B\}_{K_{bs}}$ which was formed by B during a previous legitimate run of the protocol between C and B. The attack proceeds as follows, with B and S acting exactly as in a normal run while every message which goes to and comes from the site of B is captured by C; this is possible if C is in control of the hardware communication channel of B.

1 $C \rightarrow B : \ M, A, B, \ \{N_c, M', C, B\}_{K_{cs}}$

2 $B \rightarrow S : \ M, A, B, \{N_c, M', C, B\}_{K_{cs}}, \ N_b, \{M, A, B\}_{K_{bs}}$ intercepted by C

2' $C \rightarrow S : \ M', C, B, \{N_c, M', C, B\}_{K_{cs}}, \ N_b, \{M', C, B\}_{K_{bs}}$

3 $S \rightarrow B : \ M', \{N_c, K_{cb}\}_{K_{cs}}, \{N_b, K_{cb}\}_{K_{bs}}$ intercepted by C

3' $C \rightarrow B : \ M, \{N_c, K_{cb}\}_{K_{cs}}, \{N_b, K_{cb}\}_{K_{bs}}$

4 $B \rightarrow A : \ M, \{N_c, K_{cb}\}_{K_{cs}}$ intercepted by C

At the end of this attacking run, B believes he shares the key K_{cb} with A whereas he in fact shares it with C. It may be noted that S needs to ignore the replay of M' if the attack is to succeed. This is the expected situation since M' is not intended as a nonce for any party in the protocol.

Let us now consider why the BAN logic failed to debug these two protocols.

3.1 On Protocol Idealisation

In the BAN logic the task of idealisation alters the syntax of a protocol (called the *concrete* protocol) in a number of ways: e.g. (1) some messages are simply deleted (like any plain text); (2) some are replaced by other pieces of information (e.g. a key to be distributed to principals P and Q is usually replaced with the formula $P \overset{K}{\leftrightarrow} Q$); and (3) some new pieces of information will be inserted into a protocol.

In spite of its importance, there seems no well-understood semantic rule to govern this job of idealisation. For instance, the new pieces of information that are inserted into a protocol can be terms quite freely chosen from a language with the set of BAN logical constructs as alphabet. One might argue that there cannot be formal rules to move from an informal description to a formal one. However, the lack of guidance for interpreting protocols makes this job a very expert one, so expert that even the authors of BAN logic failed do it properly. Taking the optimised Otway-Rees protocol which we have attacked in this

[1] It seems to us that the freshness of a combined message like $\sharp(X,Y)$ as a conclusion has little usefulness, if it cannot derive the freshness of an atomic message such as a key being distributed.

paper as an example, the idealised message 3 due to the authors of BAN logic is (see [2])

3 $S \rightarrow B : \cdots, \{N_b,\ A \overset{K_{ab}}{\leftrightarrow} B,\ A|\sim (M, A, B)\}_{K_{bs}}$

However, our attacking run shows that the server S has never told any such thing to B because he actually does not read the syntactic specification of a protocol. What the server can read is messages in a *run*, i.e. an *instance*, of the protocol. In the case of our attacking run, he reads A as C; thus, according to the idealisation scheme of BAN logic, the idealised message 3 for the attacking run should be

3' $S \rightarrow B : \cdots, \{N_b,\ C \overset{K_{cb}}{\leftrightarrow} B,\ C|\sim (M', C, B)\}_{K_{bs}}$

though the authors of BAN logic failed to take it into account. Indeed, it is difficult to take this line into their idealisation account, essentially as difficult as, if not more difficult than, finding the bug by informal means.

We believe that the idealisation scheme of BAN logic is fundamentally flawed. The flaw is apparent if we view a protocol specification as analogous to a program specification written in a programming language (e.g. Pascal). The specification only contains *formal parameters*, such as principals, nonces, etc. These formal parameters will be filled with *real values* during the time of run. As inputing a wrong value into a program can result computational mistakes, running a protocol by filling it with wrong principals or wrong nonces can establish false statements as long as the protocol contains statements about formal parameters, just like an idealised protocol under the scheme of BAN logic.

An extension of the BAN logic due to Gong, Needham and Yahalom [9] (GNY) resorted to the same idea of BAN idealisation, where a receiver of a message needs to "convey" a statement of formal parameters from the message. Thus, the logic of GNY will also approve the optimised Otway-Rees protocol in the same way as BAN does.

In our approach to be proposed later in this paper, protocols are also needed to be pre-processed in a way of syntactical rewriting before their analyses. As will be clear through our presentation, the nature of our rewriting can be thought of as to interpret the implicit *context-dependent* information of protocols into that of explicitly specified one (e.g. the authentication semantics reflected by the relationship between challenges and responses). Feasible guideline for the syntax rewriting will be given.

3.2 On Belief

In BAN logical constructs there are elements $P|\sim X$ and $P \triangleleft X$. These two constructs play important roles in protocol analysis because they describe the communication nature of protocols. It is worth noticing that in these logical constructs, X can be a logical formula. This is so at least in the case of the nonce-verification rule:

$$\frac{P|\equiv \sharp(X),\ P|\equiv Q|\sim X}{P|\equiv Q|\equiv X}$$

where X is clearly a formula since it appears in the conclusion as an object of a belief formula. Thus, e.g. due to the formula $Q|\sim X$, the nonce-verification rule is meant to formally promote what a principal says to what he believes. In other words, a principal positively establishes a belief due to a statement made by another principal disregarding the fact that the latter may be cheating.

Needless to say, authentication protocols which exploit cryptographic techniques work through communication of challenges and responses between principals. Such challenges and responses are designed for the purpose of verifying whether a principal under authentication is able to perform expected encipher/decipher actions within a short period of time (the principle of *peer-entity authentication* [5]). It is important to understand that any message in a protocol should *not* be interpreted as a logical statement with truth values. Here, what is important is to judge whoever has uttered a message, rather than the truth value of that message. To be concrete, assume that P believes that $Q|\sim X$, where X is a statement "*I am not Q*"; Q may still be identified (authenticated) as Q even if he said this, just as he may turn out to be someone else even if X is "*I am Q.*" As a principle of peer-entity authentication and in terms of intuition, until a secure channel has been established, any statement which is received through the channel should not be taken with any credibility.

The lack of intuition behind the scenario of BAN logic belief is also reflected by the nonce-verification rule the other way round. In the rule, the place holder X may also be filled with a nonce. Thus, a statement "P believes that Q believes a nonce" can be drawn, regardless of its senselessness. For a concrete example of a principal believing a nonce, see e.g. page 17 of [2], where $A|\equiv B|\equiv N_c$ is deduced in the analysis of the Otway-Rees Protocol.

In our approach information will be type-matchedly organised. The type match will disallow a principal to

say/see a statement, or to believe a nonce.

3.3 On Protocol Assumptions

The method of analysis in the BAN approach is to start with a set of assumptions. Many of these assumptions are obviously reasonable and necessary, such as beliefs in good shared keys between servers and other principals. However others are not so. As in protocol idealisation, it is often seen to be an expert task to decide what the initial assumptions are. No doubt an acquired intimacy with a particular protocol will make assumptions easier to spot, but this does not equate well with the aim of making protocol analysis formal.

The conclusion of an analysis may depend critically on the assumptions and so it is important that they are reasonable. There is no way of knowing in the BAN approach if slightly different assumptions would change an unsatisfactory protocol into a good one. On the other hand it also cannot be shown that the assumptions used for a good protocol are necessary, or are the weakest possible.

In our approach we overcome this problem by starting from what are the desired conclusions of a protocol and moving backwards until all the weakest and necessary pre-conditions are discovered. In some cases this will be implied by alternative sets of prior assumptions, allowing the most reasonable to be chosen for a particular application.

3.4 On Confidentiality

It is a usual assumption for peer-entity authentication through the use of cryptographical technique that prior to a protocol run a good key is already shared between certain communication parties. (In the case of a key-distribution protocol where a key-distribution server is needed, such keys are called *terminal keys*.) These pre-shared keys can be regarded as good without need to resort to any computational argument. Note that such an assumption does not apply to a *session key* which is distributed via a protocol run. The goodness of a session key should be worked out through logical reasoning computation. Thus, a mechanism to allow for such a computation is necessary. The computation should involve reasoning about beliefs about whether a session key under distribution is *exclusively* delivered to an expected set of principals. This is the vital issue of *confidentiality*.

BAN logic does not supply any mechanism to allow for such a computation. Indeed, there is no formal definition of a good session key. Informally, the goodness of a session key rests on a statement issued by a trustworthy (server) principal. The lack of the notion of confidentiality can lead to the approval by the BAN analysis of protocol which give away secrets to attackers. This is precisely what occurs in Nessett's example. Although this example is easily seen to be flawed by casual inspection, it is a cause for worry that a formal analysis can fail on such a simple protocol. As has been pointed out before [15] it could well be that more complex protocols exhibit the same problem but in a fashion that is much harder to see by informal inspection.

An extension of the BAN logic due to Kailar and Gligor [10] has also addressed the issue of confidentiality. However that approach seems to us rather restrictive as it has a number of non-intuitive features. In particular it is assumed that messages always arrive at their destination once they have been sent. This does not seem reasonable since many attacks on protocols, including that described above, involve the attacker controlling the communications of a principal. We overcome the need for such a condition by defining confidentiality in terms of the set of principals who do *not* see a message, rather than those who do.

4 A New Logic of Belief

In the previous section we have identified a number of flaws in BAN logic. Nonetheless, the logic's endeavour in applying formal methods indicates a valuable direction for the analysis of security protocols. Similar to the situation in the area of program correctness verification where application of formal methods has successfully led to computer-aided techniques, it is expected that a BAN style logic has the potential for reasoning in a computer-aided fashion. In this section a new logic is presented which adopts the basic notational framework of BAN logic, but takes a more formal approach. Considerations are taken to tackle the four major defects of BAN logic that we have discussed in the previous four subsections.

We do not regard our new technique as being more complex than the BAN logic. Although a small number of new constructs are introduced, there are in addition some simplifications. In particular the process of idealisation is far more straightforward.

4.1 Formulas

The examination that we conducted in Subsection 3.2 revealed that in BAN logic, information is handled without care of type. Examples of undesirable features due to type mismatch include cases such as when a principal believes a nonce or sees/says a statement. To eliminate these features, new logical formulas will be constructs of well-type-matched information. Three types will be used; they are $\mathcal{P}$: *principals*, $\mathcal{M}$: *messages* and $\mathcal{F}$: *formula*. As a convention, when not explicitly specified, letters $A, B, P, Q, ...$ are type of $\mathcal{P}$, letters $K, M, N, ...$ are that of $\mathcal{M}$ and $X, Y, Z, ...$, of $\mathcal{F}$. It is worth pointing out the conceptual difference between a message in type $\mathcal{M}$ and a "protocol message"; the latter is a customary term for a line in a protocol.

New logical formulas are constructed using a number of operators which are defined below.

$\_,\_$: $(\mathcal{M} \times \mathcal{P})$ *or* $(\mathcal{P} \times \mathcal{M}) \rightarrow \mathcal{M}$; operator "," is a combinator for combining a message with a principal; the result is a *message*. Notice that (M, P) and (P, M) are regarded as the same message.

$\_\Re\_$, $\_|\_$: $\mathcal{M} \times \mathcal{M} \rightarrow \mathcal{M}$; operators "$\Re$" and "|"are message combinators; $M \Re N$ and $M|N$ are combined *messages*; these two operators are proposed for comprehending the context-dependent authentication semantics; the semantics will be described in Subsection 4.2.

$\_ \wedge \_$: $\mathcal{F} \times \mathcal{F} \rightarrow \mathcal{F}$; symbol "$\wedge$" is a boolean logical conjunctor; *formula* $X \wedge Y$ is the boolean conjunction of sub-formulas X and Y; this formula is true if and only if both X and Y are true.

$\_ \models \_$: $\mathcal{P} \times \mathcal{F} \rightarrow \mathcal{F}$; *formula* $P \models X$ denotes that the *principal* P believes the *formula* X to be true.

$\_ \overset{\_}{\sim} \_$: $\mathcal{P} \times \mathcal{M} \times \mathcal{M} \rightarrow \mathcal{F}$; *formula* $P \overset{K}{\sim} M$ denotes that the *principal* P says the *message* M by using the encipherment key K; sometimes $P \sim M$ is written to mean a situation where there is no concern whether P conducted any encipherment.

$\_ \overset{\_}{\triangleleft} \_$: $\mathcal{P} \times \mathcal{M} \times \mathcal{M} \rightarrow \mathcal{F}$; *formula* $P \overset{K}{\triangleleft} M$ denotes that the *principal* P sees the *message* M by using the decipherment key K; sometimes $P \triangleleft M$ is written to mean a situation where there is no concern whether P conducted any decipherment.

$\_ \overset{\_}{\leftrightarrow} \_$: $\mathcal{P} \times \mathcal{M} \times \mathcal{P} \rightarrow \mathcal{F}$; *formula* $P \overset{K}{\leftrightarrow} Q$ denotes that the two *principals* P and Q use the *message* K as a good shared encipherment key.

$\overset{\_}{\mapsto} \_$: $\mathcal{M} \times \mathcal{P} \rightarrow \mathcal{F}$; *formula* $\overset{K}{\mapsto} Q$ denotes that the *principal* Q uses the *message* K as its good public key; its matching secret key which is denoted by K^{-1} is not known to other principals.

$sup(\_)$: $\mathcal{P} \rightarrow \mathcal{F}$; *formula* $sup(P)$ denotes that the *principal* P is a super-principal.

$\sharp(\_)$: $\mathcal{M} \rightarrow \mathcal{F}$; *formula* $\sharp(M)$ denotes that the *message* M is fresh in that it has never been seen before the current protocol run.

$\_ \triangleleft \| \_$: $2^{\mathcal{P}} \times \mathcal{M} \rightarrow \mathcal{F}$; *formula* $\mathcal{S}^C \triangleleft \| M$ denotes that *principals* in the set $\mathcal{S}^C$ cannot see the *message* M, where symbol $\mathcal{S}^C$ denotes the complement set of the set $\mathcal{S}$; this operator provides the basis for the logic to reason about confidentiality.

4.2 Protocol Message Idealisation

A key distribution protocol invariably involves peer-entity authentication of communicating parties. Authentication is achieved through exchanging message *challenges* and *responses*. (A timestamp can also play the role of a replied challenge, being an element that can be checked by the recipient.) In general, messages in a protocol can be classified into two categories with respect to these two natures. The organisation of challenging and responding messages creates a *context-dependent* relationship between such messages. This relationship determines the so-called *authentication semantics* which forms an important part of protocol behaviour.

However, the context-free syntax of a protocol specification is unable to explicitly describe the context-dependent relationship between messages. In order to take the authentication semantics into account in protocol behaviour analysis, it is necessary to interpret the implicit context-dependent information into explicit specification. The process of the interpretation is referred to as *protocol message idealisation*.

In this subsection, rules for message idealisation are given. Owing to the fact that there can be unlimited variety to design a protocol, it seems unlikely to implement a mechanical method for conducting the idealisation. Nevertheless, we believe that our rules are formally feasible, which means that, with limited human intervention, they form a guideline to correctly comprehend the authentication semantics of a security

protocol. This seems to be analogous to the situation of term rewriting in an algebraic equational system, where formally specified rewriting rules are useful although an equational rewriting system is in general semi-decidable.

First, a number of terms are to be formalised.

Definition 4.1 Atomic message, Challenge, Response and Nonsense.

Atomic message. *A piece of data in a protocol is said to be an* atomic message*, if its type is $\mathcal{M}$ and none of the symbols "," "|" "$\Re$" "(" ")" "{" "}" appears in its construction.*

Challenge. *An* atomic message *is said to be a* challenge*, if it is not a timestamp and satisfies the following condition:*

> *it (respectively, it and a function of it) appears in different lines of a protocol such that one and only one principal is entitled to* originally *send it out in one line and* subsequently *receive it (respectively, a function of it) in another line; that principal is said to be the* originator *of the challenge.*

Replied challenge. *A challenge is said to be a* replied *one, when it appears in a line which is sent to the originator of the challenge.*

Response. *An* atomic message *is said to be a* response *with respect to a challenge, if it satisfies the following conditions:*

i) it is not a timestamp, and

ii) it appears in a protocol line, where itself and a replied challenge are understood to be integrated together by the sender of that line.

Nonsense. *An* atomic message *is said to be a* nonsense*, if in no place in a protocol it is a challenge or a response, or a timestamp.*

Example 1. Assume that principals A and B share key K; also assume that T_a, T_b are timestamps and N_a, N_b are nonces of A and B, respectively. Then, according to Definition 4.1, in protocol

1. $A \rightarrow B: \ A, M, N_a, T_a$
2. $B \rightarrow A: \ B, N_b, \{N_a, T_b, K'\}_K$
3. $A \rightarrow B: \ \{N_b\}_{K'}$

N_a and N_b are challenges; their originators are A and B respectively; N_a in message 2 is a replied challenge and so is N_b in message 3; in message 2, K' is a response with respect to the challenge N_a; M is a nonsense in the protocol; finally, T_a, T_b are not nonsense; they are timestamps. Notice that N_b will not be determined as a response in message 2 because it is not integrated with any replied challenge.

Rule 4.1 Rules for Protocol Message Idealisation:

1) any nonsense is deleted;

2) if an atomic message in a line is not only a challenge but also a response then it is treated as a response in that line;

3) challenges which are separated by commas are combined using operator "|";

4) responses which are separated by commas are combined using operator "|"; the combination is called a combined response;

*5) a response with respect to a challenge is combined with that challenge using operator "$\Re$" into "*response $\Re$ replied challenge*"; the combination is still called a combined response;*

*6) a message and a timestamp, which are understood to be intergrated in a line by the sender of that line, is combined using operator "$\Re$" into "*message $\Re$ timestamp*".*

Example 2. Applying Idealisation Rules 4.1, the protocol in Example 1 will be idealised into the following:

1. $A \rightarrow B: \ A, N_a \Re T_a$
2. $B \rightarrow A: \ B, N_b, \{K' \Re N_a \Re T_b\}_K$
3. $A \rightarrow B: \ \{N_b\}_{K'}$

Notice that by deleting all nonsense from a protocol we do not mean that nonsense may not be useful information, but merely not consider it to be useful in our logical analysis.

To this end, we would like to summarise the theme of our approach to protocol message idealisation. Through the idealisation, vaguely implied context-dependent relationship between challenge and response in a concrete protocol is explicitly expressed into the form of response $\Re$ challenge $\Re$ timestamp. The process of protocol analysis is then able to make use of the information, which plays an important role in providing protocols with authentication semantics.

Finally, we point out that, in order to determine the integrity that a response with respect to a challenge or a timestamp, an effective data-origin authentication measure is crucially required. This seems to

be where human intervention takes place in this idealisation scheme. Compared with the analysis of a whole protocol, this is a much-reduced, hence simplified, problem. Furthermore, through the process of idealisation, these simplified problems are clearly singled out for extra attention.

4.3 Inference Rules

A new set of inference rules for reasoning about logical formulas is proposed. Some rules are directly adopted from BAN logic, others are new. These rules are designed to intuitively formalise the scenario of authentication and confidentiality in real applications.

The authentication rules.

$$\frac{P \models P \overset{K}{\leftrightarrow} Q \ \wedge \ P \overset{K}{\triangleleft} M}{P \models Q \overset{K}{\mid\sim} M} \quad \text{(A1)}$$

$$\frac{P \models \overset{K}{\mapsto} Q \ \wedge \ P \overset{K}{\triangleleft} M}{P \models Q \overset{K^{-1}}{\mid\sim} M} \quad \text{(A2)}$$

These two rules are essentially the same as the respective "message-meaning rules" in BAN logic. Nevertheless, we should emphasise the authentication functionality behind them. Namely, the premise $P \overset{K}{\triangleleft} M$ in these two rules refers to a decryption algorithm run by P, in which an effective measure of *data-origin* (c.f. *peer-entity*) authentication must be enforced. This is particularly so when M is not predictable by P. In BAN logic, a perfect data encryption system is assumed to guarantee the soundness of these two rules. We, however, use a new name for these rules to remind the user to be vigilant in terms of making sure that an effective message-origin authendication functionality is enforced when applying these two rules.

The confidentiality rule.

$$\frac{P \models P \overset{K}{\leftrightarrow} Q \ \wedge \ P \models S^C \triangleleft\!\| \ M \ \wedge \ P \overset{K}{\mid\sim} M}{P \models (S \cup \{Q\})^C \triangleleft\!\| \ M} \quad \text{(C)}$$

This rule is intuitively appealing. It describes a step of state evolution caused by a communication event $P \overset{K}{\mid\sim} M$, where the state system is the belief base of the principal P. Rigorously speaking, a step of state evolution should be explicitly indicated by changing the principal name P in the conclusion into a new name, say P'. Thus the logic is maintained to be monotonic. For succinctness in documentation, the renaming is omitted from our presentation but is assumed to take place.

The nonce-verification rule. The following rule formalises the notion of timeliness.

$$\frac{P \models \sharp(M) \ \wedge \ P \models Q \overset{K}{\mid\sim} M}{P \models Q \models P \overset{K}{\leftrightarrow} Q} \quad \text{(N)}$$

Namely, a principal believes a key to be good if he uses it to encrypt a message in the current protocol run.

The super-principal rule. The following rule reflects that a key distribution server has to be trusted unconditionally, i.e. a principal believes what the server believes. [2]

$$\frac{P \models Q \models X \ \wedge \ P \models sup(Q)}{P \models X} \quad \text{(S)}$$

The fresh rule. The following rule formalises a method to judge the freshness of a message.

$$\frac{P \models \sharp(M) \ \wedge \ P \triangleleft N \Re M}{P \models \sharp(N)} \quad \text{(F)}$$

The explicitly specified context-dependent information between a challenge and a response is formally applicable in this rule. Indeed, this rule formalises a widely applied method in peer-entity authentication.

The good-key rules. The following two rules define necessary conditions for a good session key.

$$\frac{P \models \{P, Q\}^C \triangleleft\!\| \ K \ \wedge \ P \models \sharp(K)}{P \models P \overset{K}{\leftrightarrow} Q} \quad \text{(G1)}$$

$$\frac{P \models \{P, Q, R\}^C \triangleleft\!\| \ K \wedge P \models sup(R) \wedge P \models \sharp(K)}{P \models P \overset{K}{\leftrightarrow} Q} \quad \text{(G2)}$$

The second rule is applicable when a session key is distributed by a key-distribution server.

Intuitive rules. The following are a number of intuitive rules. Intuitive rules are not named, because their explicit application will sometimes be omitted and the omission will hardly cause misunderstanding.

$$\frac{P \overset{K}{\triangleleft} M}{P \triangleleft M} \quad \frac{P \overset{K}{\mid\sim} M}{P \mid\sim M} \quad \frac{P \mid\sim (M, Q)}{P \mid\sim M}$$

$$\frac{P \triangleleft M \Re N}{P \triangleleft M \ \wedge \ P \triangleleft N} \quad \frac{P \triangleleft M \mid N}{P \triangleleft M \ \wedge \ P \triangleleft N}$$

Let $\mathcal{I}$ denote the above let of inference rules. The following two belief axioms are instances of the standard KD45 system where the notion of knowledge is considered to be correspondent to the world of necessity, and that of belief, the world of possibility [11, 3]:

[2] In the BAN logic a principal is only trusted with respect to certain beliefs. We find it more reasonable to trust a server for *all* beliefs since it is usually the case that an untrustworthy server can completely undermine the security of a protocol. In any event this is not an important point and we could easily accommodate the BAN approach on this matter.

The Belief Axioms

(1) $P\models (X \wedge Y)$ if and only if $P\models X \wedge P\models Y$

(2) $P\models X \wedge X/Y \in \mathcal{I}$ implies $P\models Y$

Notice that to instantiate (2) from KD45 system, consider $X/Y \in \mathcal{I}$ to be a knowledge which implies a belief in the same form; thus, the premise of axiom (2) implies $P\models X \wedge P\models X/Y$ which in turn implies $P\models Y$.

Simply apply these two axioms and notice the confidentiality rule (C); we obtain the following derived rule:

$$\frac{\begin{array}{l} P\models Q\models Q \overset{K}{\leftrightarrow} P \wedge \\ P\models Q\models S^C \lhd\!\| M \wedge \\ P\models Q \overset{K}{\mid\sim} M \end{array}}{P\models Q\models (S\cup\{P\})^C \lhd\!\| M} \quad \text{(D1)}$$

With respect to the state evolution scenario that we have discussed when postulating rule (C), the application of this rule requires that $P \notin S$; otherwise, there is no need to apply this rule because no state evolution takes place.

In the case when the communication event which causes the state evolution is $Q \overset{K}{\mid\sim} (M, R)$, the state evolution should take the third principal R into account. Intuitively, in such situation, it is not *prudent* for P to remain in a state to believe that R cannot access the message M. Thus, we postulate the following rule.

$$\frac{\begin{array}{l} P\models Q\models Q \overset{K}{\leftrightarrow} P \wedge \\ P\models Q\models S^C \lhd\!\| M \wedge \\ P\models Q \overset{K}{\mid\sim} (M, R) \end{array}}{P\models Q\models (S\cup\{R\})^C \lhd\!\| M} \quad \text{(D2)}$$

Similarly, in order to make sense, the state evolution scenario demands that $R \notin S$ be the pre-condition to apply this rule.

4.4 The Reasoning Manipulation

The *tableau* method (a system check for validity as presented in [7]) will be applied to reason about logical formulas. With the method, in order to obtain an expected goal, we start from a conclusion which is a formula describing that goal. A step of reasoning manipulation is then to reversely (bottom-up) apply a suitable inference rule and thereby to work out a set of necessary conditions given by the premises of the applied rule. Notice that term "necessary" here means that the choice of rules for the manipulation should stick to the principle of finding the weakest possible pre-conditions. For if it is not the weakest, it is not necessary. For each necessary pre-condition, the same style of reasoning manipulation goes on.

Notice that the language generated from the inference system $\mathcal{I}\cup\{D1, D2\}$ is a finite state system. Therefore, the tableau reasoning will terminate within a finite number of manipulation steps. When terminating, two finite sets of formulas are obtained. One set involves formulas $P\models\cdots$ and we call this set *the set of initial beliefs*. The other set contains formulas such as $P \lhd \cdots$ and $P\mid\sim\cdots$; we call this set *the set of communication events*.

A protocol is considered to be sound if for every principal involved in the protocol when reasoning terminates:

1) the formulas in the set of communication events can be satisfied by a protocol run, and

2) the formulas in the set of initial believes are reasonable

5 Use of the New Logic

In this section we demonstrate the use of our new logic by analysing two protocols. The first protocol is that of Nessett. In [12], Nessett demonstrated that BAN logic cannot sensibly analyse his protocol. As will be seen however, our new logic will make an easy debugging of the protocol. The second protocol analysis is to approve a fixed version of the optimised Otway-Rees protocol.

5.1 The Nessett Protocol

The Nessett protocol is as follows:

1 $A \rightarrow B:\ \{N_a, K_{ab}\}_{K_a^{-1}}$

2 $B \rightarrow A:\ \{N_b\}_{K_{ab}}$

In [12], Nessett applied BAN logic to analyse this protocol and obtained formulas such as

$$A\models A \overset{K_{ab}}{\leftrightarrow} B \quad \text{and} \quad B\models A \overset{K_{ab}}{\leftrightarrow} B$$

though, K_{ab} cannot be a good key since everyone can obtain it by deciphering the message 1 using the public key of A. Now we formally (in other words, mechanically) analyse Nessett protocol by using our logic.

Idealisation. We assume that N_a, N_b are predictable by B and A, respectively (for example they are timestamps) otherwise the protocol only contains three

pieces of nonsense and the idealised protocol is vacuous (thus, a bad protocol for this case). Then the message 2 can be seen as a function of K_{ab} and thereby K_{ab} becomes a challenge and a replied challenge in message 2. Under such an assumption, the idealised Nessett's protocol is as follows (assuming that N_a, N_b are timestamps):

1 $A \rightarrow B:\ \{K_{ab} \Re N_a\}_{K_a^{-1}}$

2 $B \rightarrow A:\ \{K_{ab} \Re N_b\}_{K_{ab}}$

With the idealised protocol, let's try to establish, for instance, formula $A \mid\equiv A \overset{K_{ab}}{\leftrightarrow} B$.

Since no server is involved the suitable good-key rule is (G1). The application demands establishing the following two formulas:

$$A \mid\equiv \{A, B\}^C \lhd\!\| K_{ab} \quad \text{and} \quad A \mid\equiv \sharp(K_{ab})$$

To establish the first formula, the only rule applicable is (C), which requires $A \mid\equiv A \overset{K_a^{-1}}{\leftrightarrow} B$. This is obviously not reasonable. Now for B, the application of (G1) demands the establishment of $B \mid\equiv \{A, B\}^C \lhd\!\| K_{ab}$. To this end, no any rule can further be applied to find a more reasonable pre-condition for this.

5.2 An Optimised Otway-Rees Protocol

We propose the following version of optimised Otway-Rees protocol and demonstrate its soundness. We refer it to as an optimised version because compared with the original version [13, 2] it processes less encryption and it removes a heavy burden from the server in terms of preventing him from doing unnecessary checks of a quite large amount of information [1]. Our proposal is as follows.

1 $A \rightarrow B:\ M, A, B, N_a$

2 $B \rightarrow S:\ M, A, B, N_a, N_b$

3 $S \rightarrow B:\ M,\ \{B, N_a \oplus K_{ab}, K_{ab}\}_{K_{as}},\ \{A, N_b \oplus K_{ab}, K_{ab}\}_{K_{bs}}$

4 $B \rightarrow A:\ M, \{B, N_a \oplus K_{ab}, K_{ab}\}_{K_{as}}$

where symbol $\oplus$ denotes the operation of bit-wise modulo-2 addition. This operation provides a protocol specified data-origin authentication functionality. The recipient, say A, can check if the integrity of the message has been unaltered after the message leaves S. What A needs to do is to $\oplus$ his nonce N_a to the second field data of the decrypted message and check if he can reveal a data to be identical to K_{ab}. This provides a way for the recipient to predict the key K_{ab} which is otherwise a random number and difficult to authenticate its integrity.

Idealisation. Following Definition 4.1 and Rule 4.1, the idealised protocol is as follows:

1 $A \rightarrow B:\ A, B, N_a$

2 $B \rightarrow S:\ A, B, N_a | N_b$

3 $S \rightarrow B:\ \{B, K_{ab} \Re N_a\}_{K_{as}},\ \{A, K_{ab} \Re N_b\}_{K_{bs}}$

4 $B \rightarrow A:\ \{B, K_{ab} \Re N_a\}_{K_{as}}$

Notice that $\oplus$ disappears from the idealised protocol; it has fulfilled its authentication functionality by indicating that the response K_{ab} and its respective challenge have, indeed, been integrated by the server. Hence, we can use operator $\Re$ to combine them.

Reasoning Manipulation. We first establish formula $A \mid\equiv A \overset{K_{ab}}{\leftrightarrow} B$. The suitable rule is (G2) since a server is involved in this protocol. Applying that rule we get the following necessary conditions:

$$A \mid\equiv \{A, B, S\}^C \lhd\!\| K_{ab},\ \ A \mid\equiv sup(S),\ \ A \mid\equiv \sharp(K_{ab})$$

Now let's first work on pre-condition $A| \equiv \{A, B, S\}^C \lhd\!\| K_{ab}$. Looking up the inference rules, we find that the only rules applicable are (i) the confidential rule (C), and (ii) the super-principal (S). The choice of the former is abandoned because it demands establishment of $A \overset{K_{as}}{\mid\sim} K_{ab}$, which cannot be satisfied by the protocol run. The application of rule (S) leads to the following two sets of conditions:

$$A \mid\equiv S \mid\equiv \{A, B, S\}^C \lhd\!\| K_{ab} \ \wedge\ A \mid\equiv sup(S)$$

Now let's concentrate on establishing $A| \equiv\ S| \equiv \{A, B, S\}^C \lhd\!\| K_{ab}$. Clearly, the applicable rules are (D1) and (D2). The pre-conditions due to the two applications are listed below, respectively:

$A \mid\equiv S \mid\equiv S \overset{K_{as}}{\leftrightarrow} A \ \wedge\ A \mid\equiv S \mid\equiv \{B, S\}^C \lhd\!\| K_{ab} \ \wedge$
$A \mid\equiv S \overset{K_{as}}{\mid\sim} K_{ab}$ due to application of (D1)

$A \mid\equiv S \mid\equiv S \overset{K_{as}}{\leftrightarrow} A \ \wedge\ A \mid\equiv S \mid\equiv \{A, S\}^C \lhd\!\| K_{ab} \ \wedge$
$A \mid\equiv S \overset{K_{as}}{\mid\sim} (K_{ab}, B)$ due to application of (D2)

Notice that we have

$$\frac{S \overset{K_{as}}{\mid\sim} (K_{ab}, B)}{S \overset{K_{as}}{\mid\sim} K_{ab}}$$

This forms evidence that the application of (D2) does not generate the weakest possible pre-conditions. To this end, the application of (D1) is chosen. The rest of the reasoning manipulation is quite straightforward and we will not provide a step-by-step argument. A tableau tree for demonstration of the successful reasoning manipulation is illustrated in Fig. 1.

The final set of initial beliefs and that of communication events for principal A can be read from the

terminal leaves (circled nodes) of the tree; they are listed below:

$A \models A \overset{K_{as}}{\leftrightarrow} S, \quad A \models \sharp(N_a), \quad A \models sup(S),$

$A \models S \models \{S\}^C \lhd\!\| K_{ab}$ and

$A \overset{K_{as}}{\lhd} (B, K_{ab}), \quad A \overset{K_{as}}{\lhd} K_{ab} \Re N_a$

The initial assumptions are reasonable ones whereas every communication event required can be satisfied by the idealised protocol. Similar analysis can be conducted on the site of principal B to result the following initial assumptions and communication events:

$B \models B \overset{K_{bs}}{\leftrightarrow} S, \quad B \models \sharp(N_b), \quad B \models sup(S),$

$B \models S \models \{S\}^C \lhd\!\| K_{ab}$ and

$B \overset{K_{bs}}{\lhd} (A, K_{ab}), \quad B \overset{K_{bs}}{\lhd} K_{ab} \Re N_b$

Once A and B start using the session key, the second-order believes about the goodness of the key:

$$A \models B \models A \overset{K_{ab}}{\leftrightarrow} B, \qquad B \models A \models A \overset{K_{ab}}{\leftrightarrow} B$$

can be deduced by using the nonce-verification rule (N).

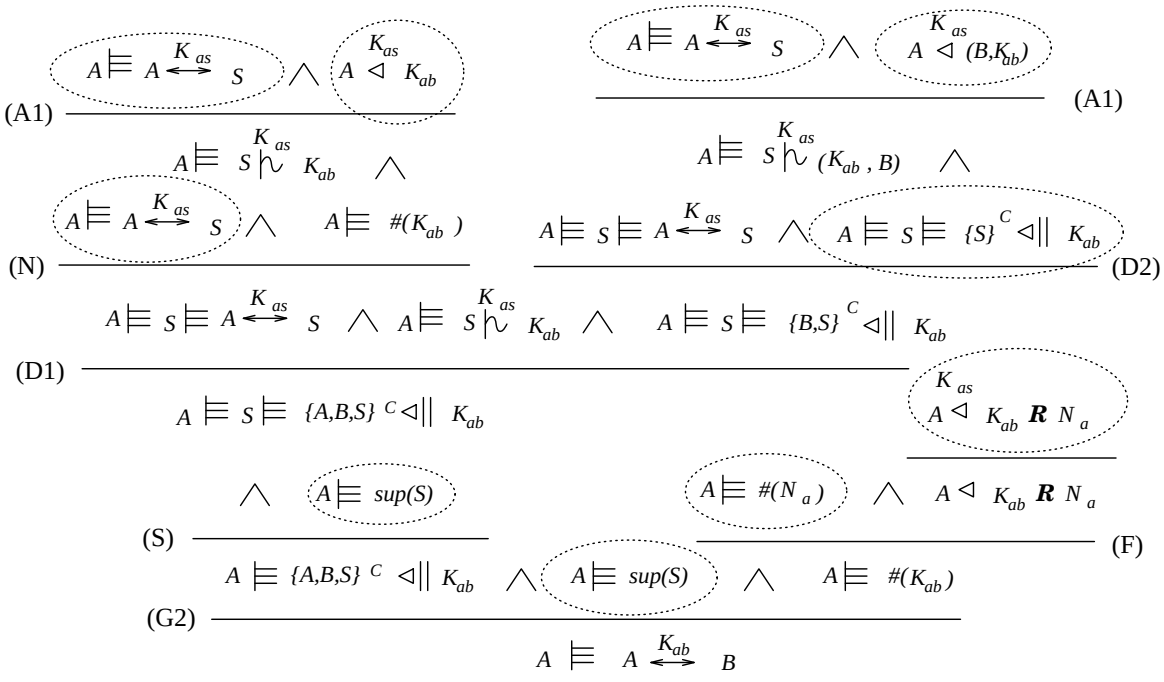

Figure 1: **A tableau for demonstrating** $A \models A \overset{K_{ab}}{\leftrightarrow} B$

6 Conclusion

We have discussed some limitations to the BAN style of logical analysis and introduced a new technique which addresses these problems. The result seems to us at once more formal, more intuitive and easier to apply. We do not pretend that our solution is final and does not have its own limitations, but we believe that it may provide a basis on which to move toward more automatic and correct protocol analysis and design.

A specific limitation is that the method is not complete, in the sense that it is possible for sound protocols not be approved by the method. Human interpretation of difficult protocols may help to solve this problem. On the other hand, since the protocol designer has unlimited variety at his disposal, and may have application dependent issues foremost in his concerns, it seems unlikely that any fixed syntax will be sufficient to handle all possible protocols. Our approach takes a step towards formal synthesis of protocols by specifying a syntax which can be handled.

There are a number of aspects which we intend to pursue further. These may involve providing a mechanism for reasoning about refutation of beliefs and developing the logic to allow for analysis of authentication protocols where there is no involvement of a server. The separation of authentication and confidentiality allows protocols with an economical mix of different cryptographic functions to be accommodated which should allow a wider variety of protocols to be analysed.

Acknowledgements We would like to thank Jeremy Jacob and anonymous referees for helpful comments.

References

[1] C. Boyd and W. Mao. Limitations of logical analysis of cryptographic protocols. In *Accepted by EuroCrypto, to appear*, 1993.

[2] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. Technical Report SRC Technical Report 39, Digital Equipment Corporation, February 1989.

[3] B.F. Chellas. *Modal Logic: An Introduction*. Cambridge University Press, 1980.

[4] P.-C. Cheng and V. D. Gligor. On the formal specification and verification of a multiparty session protocol. In *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, pages 216–233, 1990.

[5] D.W. Davies and W.L. Price. *Security for Computer Networks (second edition)*. John Wiley & Sons, 1989.

[6] J. Feigenbaum and M. Michael. Open questions, talk abstracts, and summary of discussions. In J. Feigenbaum and M. Michael, editors, *Distributed Computing and Cryptography*. a DIMACS Workshop, October 1989.

[7] M. Fitting. *Proof methods for modal and intuitionistic logics*. D. Reidel Publishing Company, 1983.

[8] K. Gaarder and E. Snekkenes. Applying a formal analysis technique to the CCITT X.509 strong two-way authentication protocol. *Journal of Cryptology*, 3(2):81–98, 1991.

[9] L. Gong, R. Needham, and R. Yahalom. Reasoning about belief in cryptographic protocols. In *Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy*, pages 234–248, 1990.

[10] R. Kailar and V. D. Gligor. On belief evolution in authentication protocols. In *Proceedings of Computer Security Foundations Workshop*, pages 171–181. IEEE Computer Society Press, 1991.

[11] L. Moser. A logic of knowledge and belief for reasoning about computer security. In *Proceedings of Computer Security Foundations Workshop II*, pages 57–63. IEEE Computer Society Press, 1989.

[12] D. Nessett. A Critique of the Burrows, Abadi and Needham Logic. *ACM Operating Systems Review*, Vol 24(No 2):35–38, April 1990.

[13] D. Otway and O. Rees. Efficient and timely mutual authentication. *Operating Systems Review*, Vol 21(1):8–10, 1987.

[14] E. Snekkenes. Exploring the ban approach to protocol analysis. In *Proceedings of Computer Security Foundations Workshop*, pages 171–181. IEEE Computer Society Press, 1991.

[15] P. Syverson. The use of logic in the analysis of cryptographic protocols. In *Proceedings of Computer Security Foundations Workshop*, pages 156–170. IEEE Computer Society Press, 1991.