Knowledge in security protocols: An operational semantics for BAN logic*

Annette Bleeker
CWI, Amsterdam
annette@cwi.nl

Lambert Meertens
CWI, Amsterdam, and
Department of Computing Science, Utrecht University
lambert@cwi.nl

0 Introduction

Communication usually aims at a certain desired knowledge change of the parties involved, rather than at a mere transport of information. In this paper, we focus on communication that takes place in the run of a protocol that is to establish a secure communication channel by means of a secret key. The protocol run must not only include the distribution of the key(s), but also convince the parties sharing the key that it can be trusted. Hence it makes sense to express the aim of such a protocol in terms of knowledge or convictions of the agents after a run of the protocol, usually under assumptions concerning what they know or believe beforehand.

Burrows, Abadi and Needham give a formalism, later named (after their creators) BAN logic, that uses modal operators of belief and that can exactly be used to reason about security properties of authentication protocols in terms of beliefs [BAN89]. Protocol designers could use it to search for failures in their design. For a protocol in which two agents, or *principals*, ask a key server to give them a key that they can share, a typical assumption would be that they believe not only the key server to be trustworthy (e.g. in being able to produce such a key), but also the channel (or key) that they each already share with it. One could then derive in BAN logic that after a run of the protocol both agents have the key and believe it is a good key.

There are, however, several implicit or informally described assumptions in BAN logic. Moreover, the semantics gets very little attention in the paper, and is mentioned only in a short, informal description. Our aim is, therefore, to construct a sound semantics for BAN, as well as a notion of correctness that makes the additional requirements for both specification and protocol explicit. But, apart from that, we would like to not only reason about the beliefs of agents, but also have some sort of justification of their belief changes during the run of the protocol. Various authors have tried to give a semantics for BAN logic or for a logic based on BAN logic, but they all stayed within the limitations of reasoning about beliefs (e.g. [GNY90, AT91]). The semantics that we present in this paper, in fact a semantics

*This paper is based on an earlier, more extensive paper *A semantics for BAN logic*, to which we refer the reader for the proofs of results presented here [BM97].

for a stronger logic than BAN logic, enables us to reason about knowledge (and, as a result, about the rightness of the participants' beliefs).

Defining a *rectify operation* that maps formulas expressing beliefs to formulas expressing knowledge — or as we call it, *right* (or *true*) beliefs — leaving other formulas intact, leads then to a theorem that expresses that principals draw the right conclusions from their beliefs. In other words: if their initial beliefs are right, their conclusions from those beliefs will be right as well.

However, logical soundness with respect to the static model does not yet establish that principals draw correct conclusions *during a protocol run*. We define a translation of a protocol run into the logic, and we also define, using operational semantics, what it means for a protocol to meet its specification, where a specification is viewed as an ordered pair consisting of assumptions and conclusions. It turns out that in order to prove that a protocol meets its specification, we need certain restrictions on the protocol, depending on the assumptions in the specification. Besides, as it turns out, those assumptions need to be of a certain form as well, in order to secure monotonicity. Those requirements can be checked statically and do not exclude well-known examples of protocols.

As we are interested in design flaws of prescribed protocols themselves, we will be focused on the analysis of a correct run of the protocols. Nevertheless, the formalism that we present may serve as a base for a broader analysis, e.g. covering failed protocol runs or runs in which participants try to cheat.

1 Language and axioms of BAN logic

The sorts we distinguish are Principal, Key, Message and Formula. There are (further unspecified) universes of constants for the sorts Principal and Key. We view (logical) formulas as being a subsort of the sort of messages, since messages can also consist of nonces, timestamps or other constants, drawn from some further unspecified universe, as well as encrypted messages. So there is an implicit injection $\mathsf{M} :: \mathsf{Formula} \longrightarrow \mathsf{Message}$.

We use variables $A, B, P, Q, R, \ldots$ for principals, Greek letters $\varphi, \psi, \ldots$ for formulas, $M, X, Y, \ldots$ for messages in general, and $K, \ldots$ for keys.

For formulas, the language of the logic has the logical constant True, the logical operators $\wedge, \vee, \rightarrow$ and $\forall$, and the operator $=$ on the sort Message. Furthermore we have the following operators:

- $(\_, \_) :: \mathsf{Message} \times \mathsf{Message} \longrightarrow \mathsf{Message}$
  (an associative, commutative and idempotent operator for message joining which is an extension of $\wedge$, the logical-and operator, so that the joining of two messages that happen to be formulas is interpreted as their conjunction[1])
- $\mathsf{believes} :: \mathsf{Principal} \times \mathsf{Formula} \longrightarrow \mathsf{Formula}$
  (we write $P\,\mathsf{believes}\,\varphi$ for what is elsewhere also known as $B_P\varphi$)
- $\mathsf{once\_said} :: \mathsf{Principal} \times \mathsf{Message} \longrightarrow \mathsf{Formula}$
  (for messages that have been uttered)
- $\mathsf{sees} :: \mathsf{Principal} \times \mathsf{Message} \longrightarrow \mathsf{Formula}$
  (for messages that have been received)
- $\_\ \mathsf{key\_of}\ (\_, \_) :: \mathsf{Key} \times \mathsf{Principal} \times \mathsf{Principal} \longrightarrow \mathsf{Formula}$
  (symmetric in the last two arguments; intuitively, $K\,\mathsf{key\_of}\,(P, Q)$ means that K is a good key between P and Q)

[1] Using the injection M mentioned above, we could write: $(\mathsf{M}\varphi, \mathsf{M}\psi) = \mathsf{M}(\varphi \wedge \psi)$.

- $\_(\_|\_)$ :: Key × Message × Principal ⟶ Message
  (for encryption; intuitively, $K(X|P)$ denotes X encrypted with K by P)
- controls :: Principal × Formula ⟶ Formula
  (an operator which is used in practice for connecting formulas with principals that exercise power over it: typically, P controls φ means that if P thinks that φ, then φ must be true, since P can either check if φ, or actively make φ true.)
- fresh :: Message ⟶ Formula
  (intuitively: fresh X if X has not been uttered before the current protocol run. Note that X remains fresh during one run.)

The "word" operators bind more tightly than the traditional logical operators, so that, e.g., P believes $\varphi \wedge \psi$ must be interpreted as $(P \text{ believes } \varphi) \wedge \psi$.

Based on the operators given in the previous section, we now give the axioms for BAN logic. In order to understand them, it is useful to know that the intended time model is very limited: there is only distinction between the present protocol run (the present) and everything that happened before that (the past). The constructs refer in principal to the present, but, as the name already indicates, once_said refers to uttering in either the present or the past (or both).

BAN1 $\vdash_{BAN} P \text{ believes } Q \text{ once\_said } (X, Y) \rightarrow P \text{ believes } Q \text{ once\_said } X$

BAN2 $\vdash_{BAN} P \text{ sees } (X, Y) \rightarrow P \text{ sees } X$

BAN3 $\vdash_{BAN} P \text{ believes } K \text{ key\_of } (P, Q) \wedge P \text{ sees } K(X|Q) \rightarrow P \text{ sees } X$

BAN4 $\vdash_{BAN} P \text{ believes } K \text{ key\_of } (P, Q) \wedge P \text{ sees } K(X|Q) \rightarrow P \text{ believes } Q \text{ once\_said } X$

BAN5 $\vdash_{BAN} P \text{ believes } \varphi \wedge P \text{ believes } \psi \rightarrow P \text{ believes } (\varphi, \psi)$

BAN6 $\vdash_{BAN} P \text{ believes } (\varphi, \psi) \rightarrow P \text{ believes } \varphi$

BAN7 $\vdash_{BAN} P \text{ believes } K \text{ key\_of } (Q, R) \rightarrow P \text{ believes } K \text{ key\_of } (R, Q)$

BAN8 $\vdash_{BAN} P \text{ believes fresh } \varphi \wedge P \text{ believes } Q \text{ once\_said } \varphi \rightarrow P \text{ believes } Q \text{ believes } \varphi$

BAN9 $\vdash_{BAN} P \text{ believes fresh } X \rightarrow P \text{ believes fresh } (X, Y)$

BAN10 $\vdash_{BAN} P \text{ believes } Q \text{ controls } \varphi \wedge P \text{ believes } Q \text{ believes } \varphi \rightarrow P \text{ believes } \varphi$

Some of the axioms may need some explanation. The third axiom states that a principal who shares a good key can decrypt messages with that key, so sees what is "inside". The fourth axiom is about authentication: if a principal believes that the shared key is good, she believes that a signed message cannot have been forged, and so must (originally) have been said by the principal which it says to be from. However, this does not mean it was necessarily said during the current protocol run, it might have been said in the past.

The eighth axiom expresses the assumption that principals are not lying, or more precisely: that they only say what they currently believe. If a message is believed to be new (for example created during this protocol run) and if it is believed to be uttered by a certain principal (for example on the grounds of the fourth axiom), then that principal is believed to believe what she uttered, since she is supposed to have believed it while uttering, and because of monotonicity she won't have changed her beliefs on this.

The last axiom expresses a form of trust: if principal P believes that Q controls the truth of φ, and furthermore P believes that Q himself believes φ is true, then P also believes φ is true.

2 Language extension and new axiomatisation

In BAN logic controls and fresh are primitive operators. We introduce them below as defined operators. In order to be able to reason about true beliefs, and to clarify some aspects of decryption, we moreover extend BAN logic with two new operators:

- $\mathsf{possesses} :: \mathsf{Principal} \times \mathsf{Key} \longrightarrow \mathsf{Formula}$
  (possession of a key means having the ability to decrypt messages encrypted with that key without necessarily believing that it belongs to a certain pair of principals)
- $\mathsf{rightly\_believes} :: \mathsf{Principal} \times \mathsf{Formula} \longrightarrow \mathsf{Formula}$
  (an auxiliary operator to express right beliefs)

While possesses is a new primitive operator, rightly_believes is also defined in terms of the other constructs.

Definition 1 *We define the operators* rightly_believes, controls *and* fresh *as follows:*

$$\mathsf{rightly\_believes} :: \mathsf{Principal} \times \mathsf{Formula} \longrightarrow \mathsf{Formula}$$

$$P\ \mathsf{rightly\_believes}\ \varphi := P\ \mathsf{believes}\ \varphi\ \wedge\ \varphi$$

$$\mathsf{controls} :: \mathsf{Principal} \times \mathsf{Formula} \longrightarrow \mathsf{Formula}$$

$$P\ \mathsf{controls}\ \varphi := P\ \mathsf{believes}\ \varphi\ \rightarrow\ P\ \mathsf{rightly\_believes}\ \varphi$$

$$\mathsf{fresh} :: \mathsf{Message} \longrightarrow \mathsf{Formula}$$

$$\mathsf{fresh}\ X := (\forall P, \varphi :: P\ \mathsf{once\_said}\ (X, \varphi)\ \rightarrow\ P\ \mathsf{believes}\ \varphi)$$

These definitions can, equivalently, be viewed as axiom schemes (interpreting ":=" as equivalence).

We present the axioms of the logic in a more general form than BAN logic: one can derive statements about the beliefs of principals, but also about statements that need not be believed, but are just "true". This enables us to talk about the rightness of those beliefs, as is already introduced by the above construct rightly_believes.

For the axiomatisation we need to define the *contents* of a message as the collection of submessages when encryption is transparent:

Definition 2 *The function* $cts[\![\_]\!]$ *takes a message and delivers a set of messages:*

$$\begin{array}{ll} cts[\![(X,Y)]\!] & := \{(X,Y)\} \cup cts[\![X]\!] \cup cts[\![Y]\!] \\ cts[\![K(X|P)]\!] & := \{K(X|P)\} \cup cts[\![X]\!] \\ cts[\![X]\!] & := \{X\} \text{ (otherwise)} \end{array}$$

The axiomatisation includes the standard axioms for *equational logic* with Modus Ponens — which subsumes propositional logic — and the standard rules for universal quantification, where a formula $\varphi \leftrightarrow \psi$ is treated as shorthand for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Beneath, we introduce a collection of axioms for the specific operators of our logic. Because of the presence of the Modus Ponens rule, we can replace inference rules $\dfrac{\varphi}{\psi}$ by axioms $\vdash \varphi \rightarrow \psi$.

For message joining there are all axioms of the forms $\vdash (X,(Y,Z)) = ((X,Y),Z)$, $\vdash (X,Y) = (Y,X)$ and $\vdash (X,X) = X$, and for the key_of operator $\vdash K\ \mathsf{key\_of}\ (P,Q) = K\ \mathsf{key\_of}\ (Q,P)$. Equational logic allows us to apply theorems of the form $\vdash \varphi[x] \wedge x = y\ \rightarrow\ \varphi[y]$.

Furthermore, we have:

- The *rationality rule*, which introduces a collection of axioms, one for every theorem of the logic, and corresponds to necessitation of the believes operator:
  A1 $\dfrac{\vdash \varphi}{\vdash P \,\mathsf{believes}\, \varphi}$

- Modus Ponens under the believes operator, or in other words, the standard K-axiom for beliefs:
  A2 $\vdash P \,\mathsf{believes}\, (\varphi \to \psi) \;\to\; (P \,\mathsf{believes}\, \varphi \;\to\; P \,\mathsf{believes}\, \psi)$

- Uttering a joint message implies uttering each of the parts:
  A3 $\vdash P \,\mathsf{once\_said}\, (X, Y) \;\to\; P \,\mathsf{once\_said}\, X$

- Uttering an encrypted message, signed by yourself, while you believe that the key is good, implies uttering of the encrypted message:
  A4 $\vdash P \,\mathsf{once\_said}\, K(X|P) \;\wedge\; P \,\mathsf{believes}\, K \,\mathsf{key\_of}\, (P, Q) \;\to\; P \,\mathsf{once\_said}\, X$

- Seeing a joint message means seeing each part separately as well:
  A5 $\vdash P \,\mathsf{sees}\, (X, Y) \;\to\; P \,\mathsf{sees}\, X$

- Awareness of what one sees:
  A6 $\vdash P \,\mathsf{sees}\, X \;\to\; P \,\mathsf{believes}\, P \,\mathsf{sees}\, X$

- If one sees a key statement, one possesses the key that it mentioned:
  A7 $\vdash P \,\mathsf{sees}\, K \,\mathsf{key\_of}\, (Q, R) \;\to\; P \,\mathsf{possesses}\, K$

- One can only believe that a certain key is good if possessing the key:
  A8 $\vdash P \,\mathsf{believes}\, K \,\mathsf{key\_of}\, (Q, R) \;\to\; P \,\mathsf{possesses}\, K$

- In BAN logic one can only decrypt with keys that are believed to be one's own key, since it does not have a separate notion for possession of any key. We allow for keys to be in possession also when they are not believed to be one's own, or even good; seeing an encrypted message while having the key in possession, means seeing the message itself:
  A9 $\vdash P \,\mathsf{possesses}\, K \wedge P \,\mathsf{sees}\, K(X|Q) \;\to\; P \,\mathsf{sees}\, X$

- A collection of axioms, stating that if a key is good, the only ones that use it for encryption are the owners, so if somewhere, someone sees a message that contains a part encrypted with that key, that part must have been said by the key owner who encrypted it:
  For all P, Q, R, X, Y, K such that $K(X|Q) \in \mathit{cts}[\![Y]\!]$ we have:
  A10 $\vdash K \,\mathsf{key\_of}\, (P, Q) \wedge R \,\mathsf{sees}\, Y \;\to\; Q \,\mathsf{once\_said}\, X$

It follows now that if part of a message is fresh, the whole of the message must be fresh as well: $\vdash \mathsf{fresh}\, X \;\to\; \mathsf{fresh}\, (X, Y)$. Note that the reverse does not hold, since a message can contain "old news" next to new data; the combination is fresh, but each element is not.

It can now be proved that each of the axioms BAN1 – BAN10 is a theorem under the new axiomatisation, and hence:

Theorem 3 *The logic as axiomatised in this section is stronger than BAN logic as axiomatised in Section 1.*

3 Model and semantics

We define now a model and an operational semantics. We view the environment as a system consisting of a finite collection of *principals*. We define for a principal P a *local state* as a tuple $(\mathcal{B}_P, \mathcal{O}_P, \mathcal{S}_P, \mathcal{K}_P)$, with the intuitive interpretation:

- $\mathcal{B}_P$, the set of formulas that P currently believes;
- $\mathcal{O}_P$, the set of (sub-)messages P once said;
- $\mathcal{S}_P$, the set of messages that P has seen so far;
- $\mathcal{K}_P$, the set of keys P possesses.

It is *closed* if it satisfies the following (mutually defined) closure properties, each of which corresponds directly to an axiom:

1. Principals believe every theorem of the logic:

$$(\vdash \varphi) \Rightarrow \varphi \in \mathcal{B}_P$$

2. Principals apply Modus Ponens in their beliefs:

$$(\varphi \rightarrow \psi) \in \mathcal{B}_P \wedge \varphi \in \mathcal{B}_P \Rightarrow \psi \in \mathcal{B}_P$$

3. If a principal said a combination of messages at a certain time, then that principal said each of the messages as well:

$$(X, Y) \in \mathcal{O}_P \Rightarrow X \in \mathcal{O}_P \wedge Y \in \mathcal{O}_P$$

   The reverse does not hold, since the presence of a joint message in $\mathcal{O}_P$ implies that both components were uttered (as a joint message) at the same time;

4. If a principal said an encrypted message and believes the key is good, then that principal said the contents of the encrypted message as well:

$$K(X|P) \in \mathcal{O}_P \wedge K \,\mathsf{key\_of}\, (P, Q) \in \mathcal{B}_P \Rightarrow X \in \mathcal{O}_P$$

5. If a principal sees a joint message, that principal sees each of the messages as well:

$$(X, Y) \in \mathcal{S}_P \Rightarrow X \in \mathcal{S}_P \wedge Y \in \mathcal{S}_P$$

   (Note that the reverse does not hold, since a joint message implies utterance of its components *at the same time*, i.e., within the same message.)

6. If a principal sees a message, then that principal also believes he sees it:

$$X \in \mathcal{S}_P \Rightarrow (P \,\mathsf{sees}\, X) \in \mathcal{B}_P$$

7. If a principal sees a key statement, then that principal possesses the key mentioned:

$$K \,\mathsf{key\_of}\, (Q, R) \in \mathcal{S}_P \Rightarrow K \in \mathcal{K}_P$$

8. If a principal believes that a certain key is a good key statement, then that principal must possess that key as well:

$$K \,\mathsf{key\_of}\, (Q, R) \in \mathcal{B}_P \Rightarrow K \in \mathcal{K}_P$$

9. If a principal P possesses a key K, and if P sees a message X labeled with Q and encrypted with K, then P also sees X itself:

$$K \in \mathcal{K}_P \wedge K(X|Q) \in \mathcal{S}_P \Rightarrow X \in \mathcal{S}_P$$

(Note that there is no closure property corresponding to the tenth axiom.)

The *closure* of a local state s is the least closed local state s' such that $s \subseteq s'$, where the ordering is obtained by component-wise lifting of the set ordering. Note that taking the closure only adds elements to the sets involved, and leaves an already closed local state unchanged.

A *global state* is a mapping from principals to local states (for each principal in the environment). Global states are ordered by lifting the ordering on local states. The closure of a global state s, denoted by $clo(s)$, is defined in the obvious way. The unqualified term "*state*" will, from now on, mean a *closed global* state. We consistently use the convention that for a state denoted by the variable s its local state for principal P is denoted by the tuple $(\mathcal{B}_P, \mathcal{O}_P, \mathcal{S}_P, \mathcal{K}_P)$, and likewise that for a state s' its local state for principal P is denoted by the tuple $(\mathcal{B}'_P, \mathcal{O}'_P, \mathcal{S}'_P, \mathcal{K}'_P)$

Lemma 4 *The function clo is monotonic, augmenting and idempotent:*

$$x \subseteq y \;\Rightarrow\; clo(x) \subseteq clo(y)$$

$$x \subseteq clo(x)$$

$$clo(clo(x)) = clo(x)$$

We define the relation $\models$ between states and formulas (where $s \models \varphi$ means: in state s formula φ holds) inductively on the structure of formulas as the least relation satisfying:

$s \models \mathsf{True}$;		
$s \models \varphi \wedge \psi$	if	$s \models \varphi$ and $s \models \psi$;
$s \models (\varphi, \psi)$	if	$s \models \varphi \wedge \psi$;
$s \models \varphi \rightarrow \psi$	if	$s \models \varphi$ implies $s \models \psi$;
$s \models (\forall x :: \varphi)$	if	$s \models \varphi[x \leftarrow u]$ holds for all terms u of the appropriate kind not containing unbound variables;
$s \models P\ \mathsf{believes}\ \varphi$	if	$\varphi \in \mathcal{B}_P$;
$s \models P\ \mathsf{once\_said}\ X$	if	$X \in \mathcal{O}_P$;
$s \models P\ \mathsf{sees}\ X$	if	$X \in \mathcal{S}_P$;
$s \models P\ \mathsf{possesses}\ K$	if	$K \in \mathcal{K}_P$;
$s \models K\ \mathsf{key\_of}\ (P, Q)$	if	for all $S \in \{P, Q\}$, all R, and all X, Y such that $K(X\|S) \in cts[\![Y]\!]$: $s \models R\ \mathsf{sees}\ Y$ implies $s \models S\ \mathsf{once\_said}\ X$.

The notation $\varphi[x \leftarrow u]$ above means: φ with u substituted for the free occurrences of x. If $\psi = (\forall \varphi :: \varphi)$, the whole formula ψ may be substituted for φ, and since $\varphi[\varphi \leftarrow \psi] = \psi$, the "recursion" in the definition of $\models$ is unbounded, whence the appeal above to "least relation".

Note that the relation $\models$ is not monotonic in its left argument, i.e., when $s \subseteq s'$, $s \models \mathcal{A}$ does in general not imply $s' \models \mathcal{A}$.

Theorem 5 (Soundness) *The logic as described in this article is sound with respect to the newly defined semantics.*

Corollary 6 *BAN logic is sound.*

4 Protocols

In the previous section we have proven our logic sound, so every theorem of the logic is a tautology in the model. This, however, does not by itself establish in any way the suitability of the logic as a tool for proving protocols correct. In fact, protocols were not referred to at all.

In most descriptions of security protocols, only the message passing is mentioned. It is implicit that principals are not supposed to continue the protocol if they have not received the correct messages which match the protocol specification. Hence, we define a protocol as a sequence of messages which have to be sent in the given order.

Definition 7 *A* protocol *is a finite sequence of* actions*, where each protocol action has the form $P \to Q : X$, signifying the sending of a message X from P to Q.*

The empty protocol is denoted by $\emptyset$, and the sequencing operator " ; " is used to denote concatenation of protocols.

The participants *of a protocol are the principals mentioned in its actions.*

We model actions as transitions from states to states. If in state s the action $P \to Q : X$ is performed, only the local states for P and Q will be changed. The new state s' is the least (closed global) state such that $s \subseteq s'$, $X \in \mathcal{O}'_P$ and $X \in \mathcal{S}'_Q$. From the closure properties we know that the local states for other principals stay unchanged. Formally:

Definition 8 *We define the transition function tra, mapping a state and an action to a new state, by*

$$tra(s, (P \to Q : X)) \;:=\; clo(s[\mathcal{O}_P \leftarrow \mathcal{O}_P \cup \{X\}, \mathcal{S}_Q \leftarrow \mathcal{S}_Q \cup \{X\}])$$

that is, for the action of sending a message X from P to Q, X is added to both $\mathcal{O}_P$ and $\mathcal{S}_Q$, and then the closure is taken (to make it a well-formed state).

We define the extension of tra to protocols by:

$$\begin{array}{lcl} tra(s, \emptyset) & := & s \\ tra(s, \mathcal{P}_1; \mathcal{P}_2) & := & tra(tra(s, \mathcal{P}_1), \mathcal{P}_2) \end{array}$$

Using Lemma 4 it follows immediately from the definition of *tra* that, for a fixed protocol, the transition function is augmenting:

Lemma 9 $s \subseteq tra(s, \mathcal{P})$

Now we can view a *specification* of a protocol as given by two sets of formulas, $\mathcal{A}$ (the *assumptions*) and $\mathcal{C}$ (the *conclusions*). Protocol $\mathcal{P}$ *satisfies* such a specification (notation: $\{\mathcal{A}\}\,\mathcal{P}\,\{\mathcal{C}\}$) when "$\mathcal{A}$ holds initially" guarantees that "$\mathcal{C}$ holds finally", i.e., after running $\mathcal{P}$. We extend the relation $\models$ between states and formulas to a relation between states and *sets* of formulas, as follows:

$$s \models \mathcal{F} \;:=\; \text{for all } \varphi \in \mathcal{F} :\; s \models \varphi$$

Using this, the semantics of the *specification triple* $\{\mathcal{A}\}\,\mathcal{P}\,\{\mathcal{C}\}$ is now defined as follows:

$$\models \{\mathcal{A}\}\,\mathcal{P}\,\{\mathcal{C}\} \quad \text{iff} \quad \text{for all states } s :\; s \models \mathcal{A} \text{ implies } tra(s, \mathcal{P}) \models \mathcal{C}.$$

Note that specifications may be composed: if $\models \{\mathcal{A}\}\,\mathcal{P}_1\,\{\mathcal{B}\}$ and $\models \{\mathcal{B}\}\,\mathcal{P}_2\,\{\mathcal{C}\}$, then $\models \{\mathcal{A}\}\,\mathcal{P}_1; \mathcal{P}_2\,\{\mathcal{C}\}$. Similarly, one can prove: if $\models \{\mathcal{A}\}\,\mathcal{P}\,\{\mathcal{B}\}$ and $\models \{\mathcal{A}'\}\,\mathcal{P}\,\{\mathcal{B}'\}$, then $\models \{\mathcal{A} \cup \mathcal{A}'\}\,\mathcal{P}\,\{\mathcal{B} \cup \mathcal{B}'\}$

5 Actions within the logic and correctness of protocols

Our aim now is to prove that if $\mathcal{C}$ can be derived in our logic from $\mathcal{A}$ together with (some yet-to-be-defined logic translation of) $\mathcal{P}$, then $\models \{\mathcal{A}\}\,\mathcal{P}\,\{\mathcal{C}\}$ holds. With stronger preconditions we will be able to prove that participants draw correct conclusions from correct assumptions: the run of the protocol will not "mislead" them.

As we have seen, the sending of a message is modelled by the transition function *tra*, which adds the message to the sender's set $\mathcal{O}$ of messages once-said, and to the receiver's set $\mathcal{S}$ of messages seen, as a form of message delivery. It does not affect other sets of sender and receiver, nor other principals' local states. The closure operator *clo* of section 3 ensures that both sender and receiver "notice" the event and draw their "conclusions". For the same effect on the derivation side we now define the expression of the send action as a logical formula.

Definition 10 *For an action $P \to Q : X$, i.e., the sending of a message X from P to Q, we define its* logic translation $\mathbf{T}(P \to Q : X)$ *as a (singleton) set of formulas:*

$$\mathbf{T}(P \to Q : X) \quad := \quad \{P \,\mathsf{once\_said}\, X \ \wedge \ Q \,\mathsf{sees}\, X\}$$

The extension to protocols is recursively defined:

$$\begin{array}{lcl} \mathbf{T}(\emptyset) & := & \emptyset \\ \mathbf{T}(\mathcal{P}_1;\mathcal{P}_2) & := & \mathbf{T}(\mathcal{P}_1) \cup \mathbf{T}(\mathcal{P}_2) \end{array}$$

Like in the model, we view the event of sending a message only locally: from P's point of view that means uttering the message, and for Q it means receiving (seeing) the message. We do not need to keep track of an overall, outsider's view of the event.

Lemma 11 *For all actions a: $\models \{\emptyset\}\, a\, \{\mathbf{T}(a)\}$*

We designed the logic to analyse what happens during a correct run of the protocol. As we mentioned before, this implicitly assumed that the participants (the official players in the protocol) had good intentions: they do not say things they do not believe, and they do not use someone else's key or a key they think is bad. In order to be able to set this as a condition on the semantic side, we define the notion of an action being allowed, depending on the assumptions there are. Note that the conditions only check what the *sender* may be assumed to believe or see: the receiver is not the actor of the event, and cannot refuse to receive a message. Therefore there are no conditions on the side of the receiver on if to receive a message.

Definition 12 *For a collection of predicates $\mathcal{A}$ and a protocol step $a = P \to Q : X$, the predicate $\mathcal{A}$ allows a is defined recursively with respect to the structure of the message X:*

$$\begin{array}{lcl} \mathcal{A} \textit{ allows } P \to Q : K(X|P) & := & \textit{for some } S : \\ & & \mathcal{A} \vdash P \,\mathsf{believes}\, K \,\mathsf{key\_of}\, (P, S) \\ & & \textit{and } \mathcal{A} \textit{ allows } P \to S : X \\ \mathcal{A} \textit{ allows } P \to Q : K(X|R), R \neq P & := & \mathcal{A} \vdash P \,\mathsf{sees}\, K(X|R) \\ \mathcal{A} \textit{ allows } P \to Q : (X, Y) & := & \mathcal{A} \textit{ allows } P \to Q : X \textit{ and} \\ & & \mathcal{A} \textit{ allows } P \to Q : Y \\ \mathcal{A} \textit{ allows } P \to Q : \varphi & := & \mathcal{A} \vdash P \,\mathsf{believes}\, \varphi \\ \mathcal{A} \textit{ allows } P \to Q : X & := & \textit{true (all other cases)} \end{array}$$

Longer protocols are allowed if each of the steps is allowed in the respective states:

$$\begin{array}{lcl} \mathcal{A} \textit{ allows } \emptyset & := & \textit{true} \\ \mathcal{A} \textit{ allows } (a;\mathcal{P}) & := & \mathcal{A} \textit{ allows } a \text{ and } (\mathcal{A} \cup \mathbf{T}(a)) \textit{ allows } \mathcal{P} \end{array}$$

To ensure monotonicity, we choose to restrict the assumptions to a certain type of formulas.

Definition 13 *We define* positive *formulas as the least set such that:*

φ	*is positive if*	$\vdash \varphi$;
K key_of (P, Q)	*is positive*;	
P possesses K	*is positive*;	
P believes fresh X	*is positive*;	
For other formulas φ :		
P believes φ	*is positive if*	φ *is positive;*
$(P$ controls $\varphi) \wedge \varphi$	*is positive if*	φ *is positive;*
$(P$ believes Q controls $\varphi)$ $\wedge(R$ believes $\varphi)$	*is positive if*	φ *is positive;*
P sees X	*is positive*;	
P once_said X	*is positive*;	
$\varphi \wedge \psi$	*is positive if*	φ *is positive and* ψ *is positive;*
$\varphi \vee \psi$	*is positive if*	φ *is positive and* ψ *is positive;*
$(\forall x :: \varphi)$	*is positive if*	$\varphi[x \leftarrow u]$ *is positive for all terms* u *of the appropriate kind not containing unbound variables.*

This is extended to finite sets of formulas: $\mathcal{F}$ is positive whenever the formula $\bigwedge \mathcal{F}$ is.

With the above defined restrictions on the set of assumptions and on the protocol itself, we have a soundness result for specifications:

Theorem 14 *If* $\mathcal{A} \cup \mathbf{T}(\mathcal{P})$ *positive,* $\mathcal{A}$ *allows* $\mathcal{P}$ *and* $\mathcal{A} \cup \mathbf{T}(\mathcal{P}) \vdash \mathcal{C}$*, then* $\models \{\mathcal{A}\}\, \mathcal{P}\, \{\mathcal{C}\}$.

6 Rectification of formulas

Now we get to our investigations of beliefs that happen to be true. Our question is if all beliefs that are assumed before a protocol are right, may we then conclude that all beliefs after a protocol run are also right? In other words, match the participants' conclusions the reality?

To express the idea of a set of assumptions being true beliefs, we define a rectify operation $\mathcal{R}[\![-]\!]$, which maps formulas to formulas. In particular, it maps formulas of the form P believes φ to P rightly_believes φ. It is defined as follows:

Definition 15

$$
\begin{array}{lcl}
\mathcal{R}[\![P \text{ believes } \varphi]\!] & := & P \text{ rightly\_believes } \varphi \\
\mathcal{R}[\![\varphi \wedge \psi]\!] & := & \mathcal{R}[\![\varphi]\!] \wedge \mathcal{R}[\![\psi]\!] \\
\mathcal{R}[\![\varphi \vee \psi]\!] & := & \mathcal{R}[\![\varphi]\!] \vee \mathcal{R}[\![\psi]\!] \\
\mathcal{R}[\![\varphi \rightarrow \psi]\!] & := & \mathcal{R}[\![\varphi]\!] \rightarrow \mathcal{R}[\![\psi]\!] \\
\mathcal{R}[\![\forall x :: \varphi]\!] & := & (\forall x :: \mathcal{R}[\![\varphi]\!]) \\
\mathcal{R}[\![\varphi]\!] & := & \varphi, \text{ other cases}
\end{array}
$$

Note the limited recursion, which stops whenever a formula with a "word" operator is encountered. The operation is extended to a set-to-set mapping in the usual way.

Directly from the definition on sets it follows that:

Theorem 16
If $\mathcal{A}_1 \subseteq \mathcal{A}_2$*, then* $\mathcal{R}[\![\mathcal{A}_1]\!] \subseteq \mathcal{R}[\![\mathcal{A}_2]\!]$*;*
If $\mathcal{A} \vdash \mathcal{C}$ *then* $\mathcal{R}[\![\mathcal{A}]\!] \vdash \mathcal{R}[\![\mathcal{C}]\!]$*; and*
If $\mathcal{A}$ *allows* a*, then* $\mathcal{R}[\![\mathcal{A}]\!]$ *allows* a.

For a stronger version of our specification soundness theorem we define a weaker notion:

Definition 17 *We define a collection of predicates $\mathcal{A}$ to be* safe *iff $\mathcal{A}$ positive or there exists a positive collection $\mathcal{A}'$ such that $\mathcal{A} = \mathcal{R}[\![\mathcal{A}']\!]$.*

Theorem 18
If $\mathcal{A} \cup \mathbf{T}(\mathcal{P})$ safe, $\mathcal{A}$ allows $\mathcal{P}$ and $\mathcal{A} \cup \mathbf{T}(\mathcal{P}) \vdash \mathcal{C}$, then $\models \{\mathcal{A}\}\, \mathcal{P}\, \{\mathcal{C}\}$.

From this stronger theorem now follows that a proof of a specification in the logic indeed ensures the right conclusions of the principals during the protocol.

Theorem 19
If $\mathcal{A} \cup \mathbf{T}(\mathcal{P})$ positive, $\mathcal{A}$ allows $\mathcal{P}$ and $\mathcal{A} \cup \mathbf{T}(\mathcal{P}) \vdash \mathcal{C}$, then $\models \{\mathcal{R}[\![\mathcal{A}]\!]\}\, \mathcal{P}\, \{\mathcal{R}[\![\mathcal{C}]\!]\}$.

7 Conclusion

We have presented an extension of BAN logic with an operational semantics which allows us to reason about semantics of protocol specifications that use modal operators of belief. We can furthermore prove that, under certain restrictions, true beliefs beforehand ensure true beliefs after a protocol run.

Our formalism resulted from a systematic attempt to formulate precise restrictions under which the changing beliefs of principals during a protocol run accurately reflect the changing state of affairs while communication takes place. The restrictions that emerged from our investigation are rather unelegant and complicated, the reason being that we wanted to keep them such that they can be checked statically and do not exclude well-known examples of protocols.

The semantic correctness criterium we have formulated only reflects correctness during a proper protocol run; it does not take all threats imposed by intruders and impostors (or dishonest participants) into account. It remains to be investigated what restrictions a stronger correctness criterion requires.

References

[AT91] Martín Abadi and Mark R. Tuttle. *A Semantics for a Logic of Authentication (Extended Abstract),* ACM Symposium on Principles of Distributed Computing, 1991, Montreal, Quebec, Canada.

[BAN89] Michael Burrows, Martín Abadi and Roger Needham. *A Logic of Authentication,* Report 39, Digital Systems Research Center, Palo Alto, CA.

[BM97] Annette Bleeker and Lambert Meertens. *A Semantics for BAN Logic,* Proceedings of the DIMACS Workshop on Design and Formal Verification of Security Protocols, 1997, Rutgers University, New Brunswick, NJ. `http://dimacs.rutgers.edu/Workshops/Security/program2/bleeker.ps`

[GNY90] Li Gong, Roger Needham and Raphael Yahalom. *Reasoning about Belief in Cryptographic Protocols,* IEEE Computer Society Symposium on Research in Security and Privacy, 1990.

[Ness90] Dan M. Nessett. *A critique of the Burrows, Abadi and Needham Logic,* Operating Systems Review, Vol. 24, No. 2, April 1990, pp 35-38.

[Snek91] Einar Snekkenes. *Exploring the BAN Approach to Protocol Analysis,* IEEE Computer Society Symposium on Research in Security and Privacy, 1991, Oakland, CA.

[Syv91] Paul Syverson. *The Use of Logic in the Analysis of Cryptographic Protocols,* IEEE Computer Society Symposium on Research in Security and Privacy, 1991, Oakland, CA.

[YW93] Alec F. Yasinsac and William A. Wulf. *Evaluating Cryptographic Protocols,* Report CS-93-66, CS & IPC, University of Virginia.