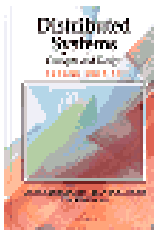

Originally published at pp. 503-9 of Coulouris, Dolllimore and Kindberg, Distributed Systems, Edition 2, 1994.

Logics of authentication

Computer security is a complex subject, and informal arguments made in an attempt to show the correctness of authentication protocols are prone to errors. Our intuition can help us understand roughly how a protocol works, but it is not normally sufficient to expose clearly all the conditions under which the protocol in fact meets the goals of authentication. When a security loophole exists, it is only a matter of time before an attacker will discover and exploit it.

The descriptions of the Needham-Schroeder and Kerberos protocols include informal arguments to the effect that the protocols achieve their stated security goals. On close examination, it can be seen, first, that the protocols are based on certain assumptions (such as: a server that knows my password is to be trusted). Secondly, it can be seen that each principal involved implicitly makes certain deductions based on the assumptions and on information it has received in messages.

Ideally, it should be possible to make explicit all the assumptions involved in a protocol, and to transform each protocol step into the application of one or more of a few general deduction rules allowing further conclusions to be drawn. To formalize an argument is to account rigorously for all the steps and assumptions made, expressing them in a symbolic way so that checking becomes a mechanical process. A logical calculus based on an agreed set of deduction rules for formally reasoning about authentication protocols is called a **logic of authentication**. The following main benefits can be derived from such a logic:

> *Correctness*: It should be possible to prove that a protocol does or does not meet its security goals. If it does not achieve the stated goals, the logic of authentication should show what it does in fact achieve.
>
> *Efficiency*: If it can be shown that the security goals can be achieved without some of the messages, contents of messages or encryptions of message contents which are part of a protocol, then the protocol can be made more efficient by eliminating them.
>
> *Applicability*: In order to judge whether a protocol can be used in a practical situation, it helps to clarify the protocol's assumptions by formally stating them. Moreover, it can be ascertained whether any of the stated assumptions are not in fact needed to achieve the authentication goals.

Burrows, Abadi and Needham describe a logic of authentication [Burrows, Abadi and Needham 1990], which is referred to henceforth as BAN logic, for the sake of brevity. BAN logic has been applied to analyse both the Needham-Schroeder and Kerberos protocols, amongst others [Burrows, Abadi and Needham 1989, Burrows, Abadi and Needham 1990]. We now introduce BAN logic, and reproduce an argument given by Burrows, Abadi and Needham [1989] that the Needham-Schroeder protocol contains an assumption that was not made explicit, but upon which the security conclusions in fact rest.

The method and formalism of BAN logic

There are three main stages to the analysis of a protocol using BAN logic. The first step is to express the assumptions and goals as *formulas* (also known as *statements*) in a symbolic notation,

so that the logic can proceed from a known state so as to be able to ascertain whether the goals are in fact reached. The second stage is to transform the protocol steps also into formulas in symbolic notation. Lastly, a set of deduction rules called *postulates* are applied. The postulates should lead from the assumptions, via intermediate formulas, to the authentication goals.

A protocol is analysed from the point of view of each particular principal *P* that participates in it. Each message received by *P* is considered in relation to previous messages received by *P* and sent by *P*. The question is what a principal should believe, on the basis of the messages it has sent and received. The assumptions of BAN logic are similar to those of the authentication protocols for whose analysis it is intended: that authentication is carried out between trustworthy principals (it will be explained below what is meant by 'trustworthy'), although attackers can attempt to foil a protocol by eavesdropping, replaying messages, or by sending malicious messages.

Belief ◊ When a principal is persuaded of the truth of a formula – or is entitled to conclude that it is true – we say that the principal **believes** it. (Bold-face words such as **believes**, **sees** and **controls** are used as predicates in the BAN formalism.) If the principal is *P* and the formula is *X*, we write *P* **believes** *X*. Note that this sense is rather different from the colloquial sense of 'believe', which does not necessarily connote 'with justification'. Thus only beliefs which are justified in terms of BAN logic are of interest. Some of these beliefs are introduced as assumptions; the others are deduced in the logic using the postulates.

We assume that trusted (trustworthy) principals do not lie about their beliefs to other principals. In other words, if *P* is trusted, and if a formula *X* is received in a message known to have been sent by *P* as part of the current run of the protocol, then it can be deduced that *P* **believes** *X*.

A principal is assumed to participate in a number of non-overlapping runs of a protocol throughout its lifetime, perhaps with different principals over time. It is important to note that, as far as time is concerned, BAN logic only has the notions of present and past, and does not assume that messages are timestamped. A principal of course knows which messages it sends during to the present run of a protocol. However, past messages – belonging to previous runs of the protocol – can be replayed any time. And what a principal **believe**d in the past is not necessarily valid in the present. For example, a server may have believed in the past that a key *K* was a secret key for use between two principals. The key might have been discovered by another principal since then.

The formulas and notation of BAN logic ◊ In order to apply the logic, the messages and the actions of the principals are transformed into formulas. P **believes** *X* is only one type of formula. The other types of formula we shall consider are as follows:

> *P* **sees** *X*: The principal *P* receives a message containing *X*. *P* might need to perform a decryption to extract *X* from the message. *P* is in a position to repeat *X* in messages to other principals. *X* can be a statement or a simple item of data such as a nonce (or a combination of both types). The term 'sees' is meant to convey the fact that the receiving principal observes *X*, but does not necessarily believe it if *X* is a statement. (In BAN logic, **see**-ing is not necessarily **believe**-ing!) Of course, messages belonging to a correct protocol should ultimately entitle principals to new beliefs, otherwise they are useless from the point of view of authentication.
>
> *P* **said** *X*: At some point in the past, *P* is known to have sent a message including *X*. This implies that, if *P* is trusted, *P* **believe**d *X* when it sent the message.
>
> *P* **controls** *X* (*P has jurisdiction over X*): *P* is trusted as an authority on *X*. For example, an authentication server is trusted as an authority on statements about the key that is allocated as a shared secret between two principals.
>
> **fresh**(*X*): *X* has not been sent in a message belonging to a previous run of the protocol. Thus nonces are values which, by definition, are constructed to be fresh.
>
> $P \overset{K}{\leftrightarrow} Q$: *P* and *Q* are entitled to use the secret key *K*. *K* is a secret between *P* and *Q* and possibly other principals trusted by *P* or *Q* (such as an authentication server).

The notation of BAN logic also includes the encryption notation used previously in this chapter: if *K* is a key, then $\{X\}_K$ means *X* encrypted with the key *K*. Finally, if *X* and *Y* are statements, then

X,*Y* means *X* and *Y*. For the sake of brevity, other constructs described by Burrows, Abadi and Needham [1990] will not be introduced here.

The postulates of BAN logic ◊ First, a point about notation. To express that the statement *Z* follows from a conjunction of statements *X* and *Y*, say, we write:

$$\frac{X,Y}{Z}$$

The main postulates – deduction rules – of BAN logic are as follows:

The message meaning rule:

$$\frac{P \textbf{ believes } P \overset{K}{\leftrightarrow} Q,\ P \textbf{ sees } \{X\}_K}{P \textbf{ believes } (Q \textbf{ said } X)}$$

We can interpret this postulate as follows: if *P* believes that it shares a secret key *K* with *Q*, and if *P* receives a message containing *X* encrypted with *K*, then *P* is entitled to believe that *Q* once said *X* (that is, that *Q* believed *X* and included *X* in a message). Note that this rule is valid only under two important assumptions. First, *X* must contain a recognisable datum to prove that the key *K* was used for encryption (and not some other key). Second, *P* must be able to tell that the message is not a replay of a message sent previously by itself, nor of one sent by any trusted principal which also knows *K*.

The nonce-verification rule:

$$\frac{P \textbf{ believes } \textbf{fresh}(X),\ P \textbf{ believes } (Q \textbf{ said } X)}{P \textbf{ believes } (Q \textbf{ believes } X)}$$

If *P* believes that *Q* once said *X*, then *P* believes that *Q* once believed *X*, by definition. But does *Q* **believe** *X* currently? The nonce-verification rule says that, if we have the additional assertion that *P* **believes** *X* is fresh, then *P* must **believe** that *Q* currently **believes** *X*. Note that *X* must not be encrypted – otherwise *Q* could merely have echoed an encrypted statement in which it does not necessarily believe.

The jurisdiction rule:

$$\frac{P \textbf{ believes } (Q \textbf{ controls } X),\ P \textbf{ believes } (Q \textbf{ believes } X)}{P \textbf{ believes } X}$$

This rule formalizes the notion of what it means for a principal to have jurisdiction over a statement: if *P* **believes** that *Q* has jurisdiction over whether or not *X* is true, and if *P* **believes** that *Q* **believes** it to be true, then *P* must **believe** in it also, since *Q* is an authority on the matter as far as *P* is concerned.

There are also various postulates for decomposing messages and for judging their freshness, for example:

(a) $\frac{P \textbf{ sees } (X, Y)}{P \textbf{ sees } X}$, (b) $\frac{P \textbf{ believes fresh}(X)}{P \textbf{ believes fresh}(X,Y)}$, (c) $\frac{P \textbf{ believes } (\text{Q } \textbf{believes}(X,Y))}{P \textbf{ believes } (\text{Q } \textbf{believes } X)}$

Informally, (a) states that a principal can observe each component of a message if it observes all of it; (b) states that a combination of components of a message is fresh if one of the components is fresh and (c) rests on our intuition that belief in a combination of several message components implies belief in them individually.

Further important postulates are given by Burrows, Abadi and Needham [1990], concerning public keys and shared secrets; but these are not necessary for this outline of the application of BAN logic to the Needham-Schroeder protocol, and they are omitted for brevity.

Applying BAN logic to the Needham-Schroeder protocol ◊ The explicit assumptions of the Needham-Schroeder protocol are as follows. These assumptions are explicit in the sense that they are mentioned or implied in the explication given in [Needham and Schroeder 1978]. However, it will shortly be seen that an extra assumption must be made in order to meet the security goals. The

same notation is used for keys and nonces as in the description of the protocol given above; in particular, principals A and B obtain a secret key through an authentication server S.

Needham-Schroeder assumptions (explicit)		
A believes:	*B believes*:	*S believes*:
$A \overset{K_A}{\leftrightarrow} S$	$B \overset{K_B}{\leftrightarrow} S$	$A \overset{K_A}{\leftrightarrow} S,\ B \overset{K_B}{\leftrightarrow} S$
S **controls** $A \overset{K_{AB}}{\leftrightarrow} B$	S **controls** $A \overset{K_{AB}}{\leftrightarrow} B$	$A \overset{K_{AB}}{\leftrightarrow} B$
S **controls** **fresh**$(A \overset{K_{AB}}{\leftrightarrow} B)$		**fresh**$(A \overset{K_{AB}}{\leftrightarrow} B)$
fresh(N_A)	**fresh**(N_B)	

The goals of the Needham-Schroeder protocol are that *A* and *B* each believe they share the secret key K_{AB}, and that moreover they each believe that the other believes it:

Authentication goals	
A believes:	*B believes*:
$A \overset{K_{AB}}{\leftrightarrow} B$	$A \overset{K_{AB}}{\leftrightarrow} B$
B **believes** $A \overset{K_{AB}}{\leftrightarrow} B$	A **believes** $A \overset{K_{AB}}{\leftrightarrow} B$

The main goals A **believes** $A \overset{K_{AB}}{\leftrightarrow} B$ and B **believes** $A \overset{K_{AB}}{\leftrightarrow} B$ are to allow A and B to communicate in private subsequently. The subsidiary goals in the last row are not general authentication goals, however, and indeed whether they are ever required is debatable. Informally, they amount to no more than that *A* and *B* each believe that the other is present, by virtue of the fact that it has used the secret key. But presence is itself a very limited notion, since it applies only to a single point in any protocol. Nothing follows from the other's presence in terms of future messages – which must all be treated with fresh skepticism.

The authentication protocol is analysed by first transforming each message into an *idealized message*, which contain only nonces and statements that are implicitly asserted by the sender of the message (if the sender is genuine, then it **believes** the statements). Note that only encrypted message contents are relevant to this analysis; clear text can provide no security and is only included in actual messages for convenience.

Message	*Idealized Message*
1. $A \rightarrow$ S: A, B, N_A	–
2. $S \rightarrow$ A: $\{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_B}\}_{K_A}$	$\{N_A, A \overset{K_{AB}}{\leftrightarrow} B$, **fresh**$(A \overset{K_{AB}}{\leftrightarrow} B)$, $\{A \overset{K_{AB}}{\leftrightarrow} B\}_{K_B}\}_{K_A}$
3. $A \rightarrow$ B: $\{K_{AB}, A\}_{K_B}$	$\{A \overset{K_{AB}}{\leftrightarrow} B\}_{K_B}$

The idealized version of Message 2 contains the statements $A \overset{K_{AB}}{\leftrightarrow} B$, **fresh**$(A \overset{K_{AB}}{\leftrightarrow} B)$ and $\{A \overset{K_{AB}}{\leftrightarrow} B\}_{K_B}$. Although these statements do not appear explicitly, *S* implicitly asserts them by sending the message in the context of the protocol. The message-meaning rule can now be applied using the first of these statements:

$$\frac{A \textbf{ believes } A \overset{K_A}{\leftrightarrow} S,\ A \textbf{ sees } \{M\}_{K_A}}{A \textbf{ believes } (S \textbf{ said } M)}$$

where $M = (N_A,\ A \overset{K_{AB}}{\leftrightarrow} B$, **fresh**$(A \overset{K_{AB}}{\leftrightarrow} B))$

Since *M* contains N_A, applying postulate (b) above gives:

$$A \textbf{ believes fresh}(M)$$

By the nonce-verification rule:

$$\frac{A \textbf{ believes fresh } (M), A \textbf{ believes } (S \textbf{ said } M)}{A \textbf{ believes } (S \textbf{ believes } M)}$$

We expand M and apply postulate (c) to obtain:

$$\frac{A \textbf{ believes } (S \textbf{ believes } (N_A,\ A \overset{K_{AB}}{\leftrightarrow} B, \textbf{ fresh}(A \overset{K_{AB}}{\leftrightarrow} B)))}{A \textbf{ believes } (S \textbf{ believes } A \overset{K_{AB}}{\leftrightarrow} B)}$$

We deduce similarly that A **believes** (S **believes fresh** ($A \overset{K_{AB}}{\leftrightarrow} B$)). Now the jurisdiction rule can be applied using the assumption that A **believes** (S **controls** $A \overset{K_{AB}}{\leftrightarrow} B$):

$$\frac{A \textbf{ believes } (S \textbf{ controls } A \overset{K_{AB}}{\leftrightarrow} B),\ A \textbf{ believes } (S \textbf{ believes } A \overset{K_{AB}}{\leftrightarrow} B)}{A \textbf{ believes } A \overset{K_{AB}}{\leftrightarrow} B}$$

And similarly:

$$\frac{A \textbf{ believes } (S \textbf{ controls fresh}(A \overset{K_{AB}}{\leftrightarrow} B)), A \textbf{ believes } (S \textbf{ believes fresh}(A \overset{K_{AB}}{\leftrightarrow} B))}{A \textbf{ believes fresh}(A \overset{K_{AB}}{\leftrightarrow} B)}$$

In summary, A **believes** $A \overset{K_{AB}}{\leftrightarrow} B$ and A **believes fresh**($A \overset{K_{AB}}{\leftrightarrow} B$). What of B? By the message-meaning rule applied to Message 3, B **believes** (S **said** $A \overset{K_{AB}}{\leftrightarrow} B$). But it cannot be deduced that B **believes** (S **believes** $A \overset{K_{AB}}{\leftrightarrow} B$) since B has no evidence that **fresh**($A \overset{K_{AB}}{\leftrightarrow} B$). Message 3 could be a replay of one sent by S in a previous run of the protocol. It is necessary to make the additional *assumption*: B **believes fresh**($A \overset{K_{AB}}{\leftrightarrow} B$). Only then can the nonce-verification and jurisdiction rules be used to deduce that B **believes** $A \overset{K_{AB}}{\leftrightarrow} B$.

Now for the remainder of the protocol. According to Burrows, Abadi and Needham [1989], the idealized messages are as follows:

Message	*Idealized Message*
4. B → A: $\{N_B\}_{K_{AB}}$	$\{N_B,\ A \overset{K_{AB}}{\leftrightarrow} B\}_{K_{AB}}$
5. A → B: $\{N_B - 1\}_{K_{AB}}$	$\{N_B,\ A \overset{K_{AB}}{\leftrightarrow} B\}_{K_{AB}}$

At Step 4, by the message meaning rule we have that A **believes** (B **said** $A \overset{K_{AB}}{\leftrightarrow} B$); and the above analysis of Steps 1–3 showed that A **believes fresh**($A \overset{K_{AB}}{\leftrightarrow} B$). By the nonce-verification rule, it can be deduced that A **believes** (B **believes** $A \overset{K_{AB}}{\leftrightarrow} B$). At Step 5 it can similarly be deduced that B **believes** (A **believes** $A \overset{K_{AB}}{\leftrightarrow} B$); the only difference is that B **believes fresh**($A \overset{K_{AB}}{\leftrightarrow} B$) is deduced from the presence of the nonce N_B.

However, it can be argued that the idealization given in Burrows, Abadi and Needham [1989] of the message at Step 4 is incorrect. This is because the message contains no recognisable text for checking that it was encrypted with the key K_{AB} as opposed to any other key. The message that A receives is merely a bit string, which could in principle represent any value N' encrypted by some key K', for which $\{N'\}_{K'} = \{N_B\}_{K_{AB}}$. The message at step 4 has no idealized form. It cannot, therefore, be deduced that A **believes** (B **believes** $A \overset{K_{AB}}{\leftrightarrow} B$). Note that this argument does not affect the idealization of Step 5: the deduction that B **believes** (A **believes** $A \overset{K_{AB}}{\leftrightarrow} B$) stands.

The foregoing illustrates the point that idealizing messages is an activity which leaves room for argument. This is an important point, since an analysis carried out using BAN logic is only as good as the informal protocol idealization upon which it rests.

Applying the BAN logic postulates can itself be tedious and is not immune to error. The postulates have been encoded for the Jape proof editor [Bornat and Sufrin 1993], which assists in

the interactive construction of proofs. In addition to automatically applying the postulates as directed by the user, Jape can suggest applications that lead to given statements. The user can therefore work back towards the assumptions from the statement to be proved, which is sometimes easier than forward reasoning.

Other formal theories of security ◊ BAN logic is not the only formal system for reasoning about security and authentication. The need for formal methods to validate the design of security systems is now widely recognized and this is an active area of research. The goal of the research is to produce methods for demonstrating that a given set of protocols and security mechanisms satisfy a required security policy, addressing the need that users have for security validation mentioned in Section 16.1.

For example, Lampson *et al.* [1992] have developed an extensive theory of authentication and trust , based on the notion of a minimal trusted computing base (TCB), in which the trustworthiness of each resource that is not included in the TCB is formally derivable. The simple notion of principal is extended to include communication channels and a **speaks for** (written $\Rightarrow$) relation between principals is introduced. A **hand-off** rule is postulated:

$$(A \textbf{ says } (B \Rightarrow A)) \supset (B \Rightarrow A)$$

enabling a principal to allow another principal to **speak for** it. This is necessary, for example, since the TCB does not include all of the communication channels required to implement any system, and it would be very inconvenient for each channel to have to be separately known to, and authenticated by, the authentication server. It is also useful when a server is required to perform an operation on behalf of a particular client (eliminating the need which might otherwise arise for the server to assume privileges greater than those of all of its clients).

A formal system called Security Logic (SL) is described by Glasgow *et al.* [1992] for reasoning about security policies – as opposed to the authentication mechanisms formalized by BAN logic. Security policies are taken to concern secrecy and integrity in a distributed system. Secrecy is formally translated into propositions about subjects (principals), and what they have permission to know; integrity is translated into propositions about what these subjects are obligated to know. An example of a proposition related to secrecy is 'Jones has permission to know the contents of Smith's files'. An example related to integrity is for a server to be obligated to know certain information as it was supplied by a set of clients.

Knowledge, permission and obligation have all been formalized in modal logics [Hughes and Cresswell 1972], and SL is in part based upon this work. Moreover, SL incorporates temporal logic [Rescher and Urquhart 1971] to describe the security properties of a system as it develops over time. Permission is similar to a safety property, and obligation is similar to a liveness property. An example of a secrecy property which a security policy might stipulate is that whenever a subject *s* knows a formula *f*, then *s* has permission to know *f*. An example of an integrity policy is that if *s* is obliged to know *f*, then *s* will eventually know *f*. Further examples are given by Glasgow *et al.* [1992].

References

Bornat and Sufrin 1993 — Bornat, R. and Sufrin, B.A. (1993). The Gist of Jape, *Tech. Report*, Oxford University Computing Laboratory.

Burrows, Abadi and Needham 1989 — Burrows, M., Abadi, M. and Needham, R. (1989). A Logic of Authentication. *Tech. Report 39*, Palo Alto CA: Digital Equipment Corporation Systems Research Center.

Burrows, Abadi and Needham 1990 — Burrows, M., Abadi, M. and Needham, R. (1990). A Logic of Authentication. *ACM Trans. Computer Systems*, vol. 8, pp. 18-36, February 1990.

Glasgow *et al.* 1992 — Glasgow, J., MacEwan, G. and Pananageden, P. (1992). A Logic for Reasoning about Security. *ACM Trans. Computer Systems*. vol. 10, no. 3. pp. 265-310.

Hughes and Cresswell 1972 — Hughes, G.E. and Cresswell, M.J. (1972). *An Introduction to Modal Logic*, University Paperbacks.

Lampson *et al*. 1992 — Lampson, B.W., Abadi, M., Burrows, M. and Wobber, E. (1992). Authentication in Distributed Systems: Theory and Practice. *ACM Trans. on Computer Systems*, vol. 10, no. 4, pp. 265–310.

Needham and Schroeder 1978 — Needham, R.M. and Schroeder, M.D. (1978). Using encryption for authentication in large networks of computers. *Comms. ACM*, vol. 21, pp. 993–9.

Rescher and Urquhart 1971 — Rescher, N. and Urquhart, A. (1971). *Temporal Logic*. Springer-Verlag.