SEMINAR REPORT

ON

BAN LOGIC

Submitted by

Name : Abhijeet Chatarjee

Roll No.: 14IT60R11

SCHOOL OF INFORMATION TECHNOLOGY

INDIAN INSTITUTE OF TECHNOLOGY,

KHARAGPUR-721302 (INDIA)

Abstract:

Authentication protocols are the basis of security in many distributed systems, and it is therefore essential to ensure that these protocols function correctly. Unfortunately, their design has been extremely error prone. Most of the protocols found in the literature contain redundancies or security flaws. A simple logical calculus based on the agreed set of deduction rules can allow us to describe the beliefs of trustworthy parties involved in authentication protocols and the evolution of these beliefs as a consequence of communication. BAN logic is one of the formal protocol verification techniques that help us to prove whether a protocol works correctly. In this report, Kerberos protocol is chosen and proved because of its practical importance for authentication.

1. Introduction

Computer security is a complex subject, and informal arguments made in an attempt to show the correctness of authentication protocols are prone to errors. The descriptions of all the authentication 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. 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 developed logic for analysing authentication protocols. The logic is called BAN-logic. With the logic all public - and shared key primitives are formalised

and also the notion of a `fresh message'. This makes it possible to formalise a challenge - response protocol.

BAN-logic can be used for answering the following questions:

- To what conclusions does this protocol come?
- What assumptions are needed for this protocol?
- Does the protocol use unnecessary actions, which can be left out?
- Does the protocol encrypt anything which could be sent in plain, without weakening the security?

The BAN logic makes it possible to reason in a simple way over cryptographic protocols in a formal way. The basis for the logic is the belief of a party in the truth of a formula. A formula does not necessarily be true in the general sense of truth. It should be kept in mind that the BAN logic is meant for reasoning over cryptographic protocols. Verification with BAN logic does not necessarily imply that no attacks on the protocol are possible. A proof with the BAN logic is a good proof of correctness, based on the assumptions. However, questions may arise over the semantics of the logic and the logic does exclude possible attacks. BAN logic has its purpose, because it can be used in the design of a cryptographic protocol. The use of a formal language in the design process can exclude faults.

In the later part of this report, we show that how the logic has enabled us to prove Kerberos protocol which is most widely used authentication protocol. Since this protocol operate at an abstract level, it does not consider errors introduced by concrete implementations of a protocol, errors such as deadlocks, or even inappropriate use of cryptosystems. This focuses on the beliefs of trustworthy parties involved in the protocols and on the evolution of these beliefs as a consequence of communication. This kind of study seems to be one of the most needed by current protocol designers; it has often uncovered subtleties and suggested improvements in existing protocols.

2. The Formalism

This section deals with the syntax and semantics of BAN logic, its rules and the transformations that is applied to a protocol

2.1 Basic Notation

In the BAN logic several sorts of objects are distinguished: principals, encryption keys, and formulas (also called statements). Typically, the symbols A, B, and S denote specific principals; K_{ab}, K_{as}, and K_{bs} denote specific shared keys; K_a, K_b, and K_S denote specific public keys, while the inverses of these keys (e.g. K_{A-1}) represents each principal's private key; N_A, N_B and N_S identify nonce's as well as their creator. The symbols P, Q, and R range over principals; X and Y range over statements; and K ranges over encryption keys.

The following constructs are used in BAN logic:

- P **believes** X: 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. Some of these beliefs are introduced as assumptions; the others are deduced in the logic using the postulates.
- 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: The principal P at some time sent a message including the statement X. It is not known whether the message was sent long ago or during the current run of the protocol, but P believed X when it sent the message.
- P **controls** X: P has jurisdiction over X. The principal P is an authority on X and should be trusted on this matter. This construct is used when a principal has delegated authority over some statement. For example, encryption keys need to be generated with some care, and in some protocols certain servers are trusted to do this properly. This may be expressed by the assumption that the principals believe that the server has jurisdiction over statements about the quality of keys.
- **fresh**(X): The formula X is fresh; that is, X has not been sent in a message at any time before the current run of the protocol. This is usually true for nonces, that is, expressions invented for the purpose of being fresh. Nonces commonly include a timestamp or a number that is used only once.
- $P \leftrightarrow Q$: P and Q may use the shared key K to communicate. The key K is good, in that it will never be discovered by any principal except P or Q, or a principal trusted by either P or Q.
- $\{X\}_K$: X encrypted with the key K.

2.2 The postulates of BAN logic:

1) *Message-meaning rule*:

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

This rule concerns the interpretation of messages. If P believes that the key K is shared with Q and sees X encrypted under K, then P believes that Q once said X. For this rule to be sound, we must guarantee that P did not send the message himself; it is assumed that $\{X\}_K$ stands for a formula of the form $\{X\}_K$ from R, and to require that R not equal to P.

2) *Nonce-verification rule:*

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

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.

3) *Jurisdiction rule:*

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

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.

4) *Decomposition rules:*
There are also various postulates for decomposing messages and for judging their freshness. 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.

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

2.3 Idealized Protocols:

In the literature, authentication protocols are described by listing their messages; each message is typically written in the form

$$\text{P} \rightarrow \text{Q: message.}$$

This denotes that the principal P sends message to the principal Q. The message is presented in an informal notation designed to suggest the bit-string that a concrete implementation would use. This presentation is often ambiguous and not an appropriate basis for our formal analysis. Therefore, we transform each protocol step into an idealized form. A message in the idealized protocol is a formula. For instance, the protocol step

$$A \rightarrow B: \{A, K_{ab}\}_{K_{bs}}$$

may tell B, who knows the key K_{bs}, that K_{ab} is a key to communicate with A. This step should then be idealized as

$$A \rightarrow B: \left\{A \overset{K_{ab}}{\longleftrightarrow} B\right\}_{K_{bs}}$$

The idealized protocols of the examples given below do not include clear text message parts; idealized messages are of the form $\{X_1\}_{K_1}, \ldots, \{X_n\}_{K_n}$. We have omitted clear text communication simply because it can be forged, and so its contribution to an authentication protocol is mostly one of providing hints as to what might be placed in encrypted messages.

3. The Goals of Authentication:

Goal of the authentication is to make a belief that the communication is being done between only trustworthy parties, without any intruder. Often the authentication focuses on establishing a session key between the server and the client.

Thus, we might deem that authentication is complete between A and B if there is a K such that

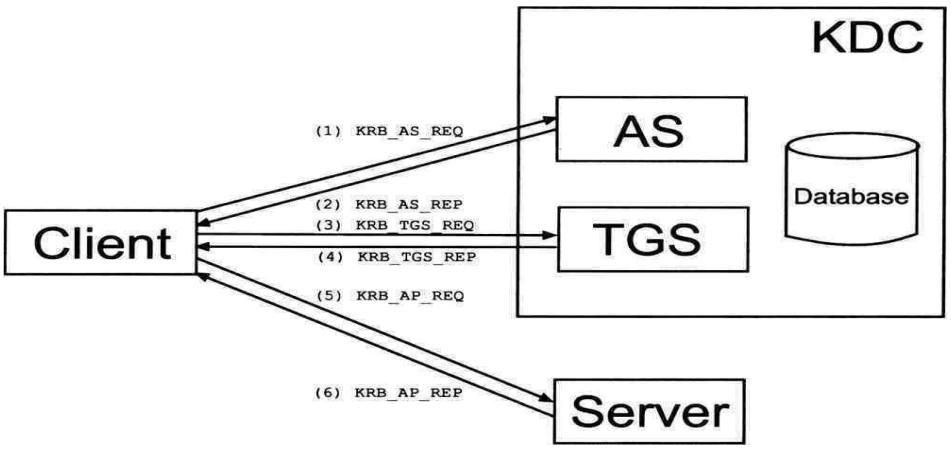
A **believes** $A \overset{K}{\leftrightarrow} B$, B **believes** $A \overset{K}{\leftrightarrow} B$

Some protocols achieve more than this, as in the following example:

A **believes** B **believes** $A \overset{K}{\leftrightarrow} B$, B **believes** A **believes** $A \overset{K}{\leftrightarrow} B$

4. THE KERBEROS PROTOCOL:

In this section we will prove Kerberos protocol using so discussed BAN logic and check the validity of the protocol for practical use. The authentication process is as follows:



In very brief, all the message exchanges are shown below:

C: Client;

KDC: Key Distribution Center, is consists on AS and TGS;

AS: Authentication Serve;

TGS: Ticket Granting Server;

V: Application Server;

K_X: X's shared key with KDC;

$K_{X,Y}$: X's shared session key with Y;

$\{m\}_K$: message m is encrypted by key K;

TGT: Ticket Granting Ticket, used to visit TGS;

$T_{X,Y}$: the Ticket that X visit Y;

$A_{X,Y}$: the authentication symbol of X and Y;

N_1, N_2 and N_C are random numbers Timestamp;

Lifetime: effective survival time;

Addr: the IP address of client C.

Message exchanges: This exchanges also illustrates idealized protocol version of Kerberos.

1) the authentication service exchange

The message exchanges (1) and (2) between the clients and AS is called original ticket exchange. This phase is the process that the client applies for the ticket TGT using to communicate with TGS and the session key from KDC. That is called authentication service exchange and called AS exchange. When the client logs into the network he needs input his user name and password. He applies for TGT from AS by sending message KRB_AS_REQ and AS responds him as message KRB_AS_REP.

a) C → AS : applying for the Ticket Granting Ticket (KRB_AS_REQ)

$$C, TGS, addr, N_1, lifetime$$

b) AS → C *:* the Ticket Granting Ticket (KRB_AS_REP)

$$\{K_{C,TGS}, T_{C,TGS}\}K_C$$

$$T_{C,TGS} = \{TGS, C, addr, N_1, lifetime, K_{C,TGS}\}K_{TGS}$$

2) *the authorization service exchange*

The authorization service exchange is also called the TGS exchange. The authorization service exchange process is consists on the message exchange (3) and (4). That is the process that client applies for the ticket $T_{C,V}$ and session key communicating with the application server V from TGS. TGS exchange is the message exchange between client and TGS and the message form is same as that in the AS exchange. But there is a significant difference that the session key but not the client's shared key is used as the encrypted and decrypted key in this process. The TGS exchange is consists of two messages that are KRB_TGS_REQ and KRB_TGS_REP.

c) C → TGS : applying for the server ticket (KRB_TGS_REQ)

$$V, N_2, \text{lifetime}, T_{C,TGS}, A_{C,TGS}$$

$$A_{C,TGS} = \{C, \text{addr}, \text{timestamp}\}K_{C,TGS}$$

d) TGS → C : server ticket (KRB_TGS_REP)

$$\{K_{C,V}, T_{C,V}\}K_{C,TGS}$$

$$T_{C,V} = \{V, C, \text{addr}, N2, \text{lifetime}, K_{C,V}\}K_{V,TGS}$$

3) *the client / application server exchange*

After AS exchange and TGS exchange client gets the ticket $T_{C,V}$ and session key $K_{C,V}$ for visiting V. The both identity authentication will be achieved after they pass the client/application server exchange. The client / application server exchange is consists of two messages which are KRB_AP_REQ and KRB_AP_REP. KRB_AP_REP is only used when there is need two-way authentication and server wants to prove its identity to client.

e) C → V : applying for service (KRB_AP_REQ)

$$V, T_{C,V}, A_{C,V}$$

$$A_{C,V} = \{C, \text{addr}, N_C\}K_{C,V}$$

f) V → C : server authentication (KRB_AP_REP)

$$\{N_{C+1}\}K_{C,V}$$

Initial assumptions:

Some reasonable assumptions constructed for the analysis condition of Kerberos protocol is as follows:

1) C **believes** $C \overset{K_C}{\leftrightarrow} AS$
2) V **believes** $V \overset{K_{V,TGS}}{\longleftrightarrow} TGS$
3) C **believes** AS **controls** $T_{C,TGS}$
4) C **believes** TGS **controls** $T_{C,V}$
5) C **believes** TGS **controls** $K_{C,V}$
6) V **believes** TGS **controls** $T_{C,V}$
7) C **believes fresh**(N_1)
8) V **believes fresh**(N_C)

9) C **believes fresh**(N_C)

Now the protocol proof using BAN logic is as follows:

1. According to message meaning rule (Rule 1),

$$\frac{\mathrm{C}\ \textbf{believes}\ \mathrm{C} \overset{K_C}{\leftrightarrow} \mathrm{AS}, \mathrm{C}\ \textbf{sees}\ \{T_{C,TGS}\}_{K_C}}{\mathrm{C}\ \textbf{believes}\ \mathrm{AS}\ \textbf{said}\ T_{C,TGS}}$$

2. By Nonce-verification rule (Rule 2),

$$\frac{\mathrm{C}\ \textbf{believes fresh}(N_1),\ \mathrm{C}\ \textbf{believes}\ \mathrm{AS}\ \textbf{said}\ T_{C,TGS}}{\mathrm{C}\ \textbf{believes}\ \mathrm{AS}\ \textbf{believes}\ T_{C,TGS}}$$

3. By rule 3,

$$\frac{\mathrm{C}\ \textbf{believes}\ \mathrm{AS}\ \textbf{controls}\ T_{C,TGS},\ \mathrm{C}\ \textbf{believes}\ \mathrm{AS}\ \textbf{believes}\ T_{C,TGS}}{\mathrm{C}\ \textbf{believes}\ T_{C,TGS}}$$

4. By rule 4,c,

$$\mathrm{C}\ \textbf{believes}\ K_{C,TGS}$$

5. Again by message meaning rule,

$$\frac{\mathrm{C}\ \textbf{believes}\ \mathrm{C} \overset{K_{C,TGS}}{\longleftrightarrow} \mathrm{TGS}, \mathrm{C}\ \textbf{sees}\ \{T_{C,V}\}_{K_{C,TGS}}}{\mathrm{C}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{said}\ T_{C,V}}$$

6. By rule 2,

$$\frac{\mathrm{C}\ \textbf{believes fresh}(T_{C,V}),\ \mathrm{C}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{said}\ T_{C,V}}{\mathrm{C}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{believes}\ T_{C,V}}$$

7. Again by rule 3,

$$\frac{\mathrm{C}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{controls}\ T_{C,V},\ \mathrm{C}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{believes}\ T_{C,V}}{\mathrm{C}\ \textbf{believes}\ T_{C,V}}$$

8. Finally according to rule 4,c again,

$$\mathrm{C}\ \textbf{believes}\ \mathrm{C} \overset{K_{C,V}}{\longleftrightarrow} V$$

9. According to message meaning rule,

$$\frac{\mathrm{V}\ \textbf{believes}\ \mathrm{V} \overset{K_{V,TGS}}{\longleftrightarrow} \mathrm{TGS}, \mathrm{V}\ \textbf{sees}\ \{T_{C,V}\}_{K_{V,TGS}}}{\mathrm{V}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{said}\ T_{C,V}}$$

10. Again by nonce-verification rule,

$$\frac{\mathrm{V}\ \textbf{believes fresh}(T_{C,V}),\ \mathrm{V}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{said}\ T_{C,V}}{\mathrm{V}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{believes}\ T_{C,V}}$$

11. By Jurisdiction rule again,

$$\frac{\mathrm{V}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{controls}\ \mathrm{T_{C,V}},\ \ \mathrm{V}\ \textbf{believes}\ \mathrm{TGS}\ \textbf{believes}\ \mathrm{T_{C,V}}}{\mathrm{V}\ \textbf{believes}\ \mathrm{T_{C,V}}}$$

12. Finally, according to rule 4,c ,

$$\mathrm{V}\ \textbf{believes}\ \mathrm{C} \overset{K_{C,V}}{\longleftrightarrow} V$$

13. By rule 1 (message meaning rule),

$$\frac{\mathrm{V}\ \textbf{believes}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}, \mathrm{V}\ \textbf{sees}\ \left\{\mathrm{N_C}, \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}\right\}_{\mathrm{K_{C,V}}}}{\mathrm{V}\ \textbf{believes}\ \mathrm{C}\ \textbf{said}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}}$$

14. By rule 2,

$$\frac{\mathrm{V}\ \textbf{believes fresh}\left(\mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}\right),\ \mathrm{V}\ \textbf{believes}\ \mathrm{C}\ \textbf{said}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}}{\mathrm{V}\ \textbf{believes}\ \mathrm{C}\ \textbf{believes}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}}$$

Finally, have derived

$$\mathrm{V}\ \textbf{believes}\ \mathrm{C}\ \textbf{believes}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}$$

Similarly, we can get

$$\mathrm{C}\ \textbf{believes}\ \mathrm{V}\ \textbf{believes}\ \mathrm{C} \overset{\mathrm{K_{C,V}}}{\longleftrightarrow} \mathrm{V}$$

which proves our goal of authentication.

5. Conclusion:

Applying the BAN logic postulates can itself be tedious and is not immune to error. We have proved Kerberos protocol using BAN logic which results it positively. But there are various attacks like replay attacks, man in middle attack, but BAN logic is unable to capture them as it talks about the abstract view of any protocol and doesn't go to the implementation and concrete view. BAN logic is the foundation of other protocol specification languages that are more expressive.

6. References:

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

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

[3] NEUMAN C. RFC 1510, The Kerberos Network Authentication Service (V5).1993.

[4] STEINER G. Kerberos, "an authentication service for open network system", Proceedings of the Winter 1988 Usenix Conference.1988.

[5] “A Logic of Authentication by Burrows, Abadi and Needham” by Kyntaja.

http://www.tml.hut.fi/Opinnot/Tik-110.501/1995/ban.html