What is BAN Logic?

- A formal rule system for reasoning about agent beliefs:
  What are the valid beliefs that agents accumulate as a protocol run proceeds?
- BAN logic supports reasoning about timeliness: agent beliefs expire with time.

What are Questions that BAN Logic Aims to Answer?

- What does this protocol achieve?
- Does this protocol need more assumptions than another one?
- Does this protocol do anything unnecessary that could be left out without weakening the protocol?
- Does this protocol encrypt something that could be sent in clear without weakening the protocol?

What are Limitations of BAN Logic?

- BAN logic assumes that agents never publish secrets, but BAN logic does not verify this.
- BAN logic assumes that agents can recognize type flaws, but BAN logic does not verify the absence of type flaws.
  - In particular, BAN logic assumes that agents always recognize and ignore messages that they have sent themselves.
- BAN logic assumes that all protocol participants are honest. No compromised agents are considered. Attackers do not have valid keys.
- Like always in this class, BAN logic assumes perfect cryptography.
BAN Logic’s Model of Time

- Time is divided into two epochs: the present and the past.
- The present begins at the start of the particular protocol run that is under consideration.

Formulas: Syntax Domains

Syntax domains.

- A set of agent identifiers. \(A, B\) range over agents.
- A set of key identifiers. \(k\) ranges over keys.
- A set of identifiers for atomic formulas. \(n\) over these.
- A set of formulas. \(X, Y\) range over formulas.

- Some formulas represent boolean expressions, i.e., formulas that can be either true or false. E.g.:
  - “\(A\) believes that \(k\) is a shared key of \(A\) and \(B\)”
  - Formally: \(A \leftrightarrow k \rightarrow B\)
- Other formulas represent idealized messages. E.g.:
  - \(\{na, A \leftrightarrow k \rightarrow B\}_kas\)
- Both boolean expressions and idealized messages are drawn from the same single syntax domain of formulas.
- Atomic formulas typically represent nonces.

Formulas: Basic Formulas

- \(A \leftrightarrow k \rightarrow B\): \(k\) is a key shared by \(A\) and \(B\).
- \(\text{fresh}(X)\): \(X\) is a valid belief in the present epoch.

- In \(\text{fresh}(X)\), the argument \(X\) could for instance be a nonce \(n\) or a formula of the form \(A \leftrightarrow k \rightarrow B\):
  - Intuitively, in these cases \(\text{fresh}(X)\) holds iff \(n\) (resp. \(k\)) has been generated in the present epoch.
- The original BAN papers:
  - write \#(\(X\)) instead of \(\text{fresh}(X)\).

Formulas: Idealized Messages

Formulas (cont.)

- \(X_1, \ldots, X_n\): concatenation
- \(\{X\}_k\): \(X\) encrypted with \(k\)
Formulas: Receives (aka Sees), Once Said

• A receives X : A receives idealized message X
• A said X : A once sent idealized message X

• We pronounce:
  - A said X as “A once said X”.
  - This emphasizes that A may have sent X long ago.

• The original BAN papers:
  - write A ◐ X instead of A receives X,
  - pronounce “A sees X” instead of “A receives X”,
  - write A ∼ X instead of A said X.

Formulas: Belief, Jurisdiction

• A believes X : A believes X
• A controls X : A has jurisdiction over X

• The next sentence is provable in BAN logic:
  - If A believes B controls X and A believes B believes X, then A believes X.
  - B could, for instance, have jurisdiction to generate shared keys for A and B. (Then X would be $A\leftrightarrow kab, B$.)

• The original BAN papers:
  - write $A\equiv X$ instead of A believes X,
  - write $A\Rightarrow X$ instead of A controls X.

Typical Protocol Goals

A protocol that establishes a session key $kab$ for A and B typically has the goal that at the end of a successful run the following statements hold:
• A believes $A\leftrightarrow kab, B$
• B believes $A\leftrightarrow kab, B$

Sometimes A and B also want to know that the other agent knows about the key:
• A believes B believes $A\leftrightarrow kab, B$
• B believes A believes $A\leftrightarrow kab, B$

Some key establishment protocols are weaker and only achieve that A believes that B was “in the loop” during this protocol run:
• A believes B believes X for some X
• B believes A believes X for some X

Rules: The Receives Rule

\[
\text{(Receives)} \quad \begin{array}{c}
A \text{ receives } \{X\}_k \\
A \text{ believes } A \leftrightarrow kab, B
\end{array} \quad \Rightarrow \quad B \text{ believes } B \text{ said } X
\]

• This rule should be pronounced as:
  - If A receives $\{X\}_k$ and A believes $A \leftrightarrow kab, B$, then A believes B said X.

• The other proof rules are all of this format, too.
• This rule is sound under the assumption that each agent recognizes and ignores his own messages.
Rules: The Fresh Rule

\[(\text{Fresh})\]
\[
\begin{array}{c}
A \text{ believes } B \text{ said } X \\
A \text{ believes fresh } (X)
\end{array}
\]
\[
A \text{ believes } B \text{ believes } X
\]

Rules: The Jurisdiction Rule

\[(\text{Jurisdiction})\]
\[
\begin{array}{c}
A \text{ believes } B \text{ believes } X \\
A \text{ believes } B \text{ controls } X
\end{array}
\]
\[
A \text{ believes } X
\]

Rules: The Fresh Inject Rule

\[(\text{Fresh Inject})\]
\[
A \text{ believes fresh } (X_i)
\]
\[
A \text{ believes fresh } (X_1, \ldots, X_i, \ldots, X_n)
\]

\text{This rule says:}
\begin{itemize}
  \item If \(A\) believes that some part \(X_i\) of a formula is fresh, then she believes that the whole formula is fresh.
  \item Typically, \(X_i\) is a nonce \(n\), a freshly generated session key \(k\), or a basic formula of the form \(A \leftrightarrow B\) where \(k\) is freshly generated.
\end{itemize}

Rules: Selection Rules

\[(\text{R-Select})\]
\[
A \text{ receives } (X_1, \ldots, X_i, \ldots, X_n)
\]
\[
A \text{ receives } X_i
\]

\[(\text{BB-Select})\]
\[
A \text{ believes } B \text{ believes } (X_1, \ldots, X_i, \ldots, X_n)
\]
\[
A \text{ believes } B \text{ believes } X_i
\]

\[(\text{BS-Select})\]
\[
A \text{ believes } B \text{ said } (X_1, \ldots, X_i, \ldots, X_n)
\]
\[
A \text{ believes } B \text{ said } X_i
\]
Protocol Analysis with BAN Logic

Given an idealized protocol narration and a set of formulas $Init$, which represents the initial assumptions:

$$[Init]$$

\[
\begin{align*}
A_1 & \to B_1 : X_1 \\
\vdots \\
A_n & \to B_n : X_n
\end{align*}
\]

- We say that $Y$ holds after $(A_i \to B_i : X_i)$, whenever $Y$ can be derived, using the BAN rules, from the following assumptions:
  - $Init$, $B_1$ receives $X_1$, ..., $B_i$ receives $X_i$
- We say that $Y$ holds before $(A_i \to B_i : X_i)$, whenever $Y$ holds after $(A_{i-1} \to B_{i-1} : X_{i-1})$.

Example: Needham–Schroeder Symmetric Key

- **Goal:** $A$ and $B$ want to establish a short-term session key $kab$.
- They trust the server $S$ to generate session keys.
- They share long-term keys $kas$ and $kbs$ with $S$.

1. $A \to S : A, B, na$
2. $S \to A : \{msg_2, na, B, kab, \{msg_3, A, kab\}_{kbs}\}_{kas}$
3. $A \to B : \{msg_3, A, kab\}_{kbs}$
4. $B \to A : \{msg_4, nb\}_{kab}$
5. $A \to B : \{msg_5, nb\}_{kab}$

Protocol Analysis with BAN Logic (cont.)

Conceptually, protocol analysis proceeds in the following four steps.

1. Idealize the protocol by replacing concrete messages by idealized messages.
2. State the initial assumptions $Init$.
3. For each event $(\cdot \to B : X)$, check that protocol goals that are supposed to hold after this event really hold. Typically, these goals are of the form $(B \text{ believes } \cdot)$.
4. For each event $(A \to \cdot : X)$ and all boolean expressions $Y$ that occur in $X$ under an encryption by $A$, check that $(A \text{ believes } Y)$ holds before the event.

In practice, protocol analysis is often an iterative process where the intended protocol goals and the initial assumptions get discovered during the analysis.

Idealizing Protocols

- Drop unencrypted messages, because they do not contribute to the growth of agent beliefs.
- Drop all other data that does not contribute to the growth of agent beliefs. (Usually only nonces and timestamps need to be left in.)
- In idealized messages, include formulas representing beliefs that get conveyed from the sender to the receiver.
- Of course, senders should believe all formulas that they convey. (Double-check this later, using BAN-logic.)
Idealizing NSSK

Show that the Idealization of Message 2 is Sound

Assume: \textit{Init}
Show: S believes A $\xrightarrow{kab} B$ and S believes fresh(A $\xrightarrow{kab} B$)
That's easy.

| S believes A $\xrightarrow{kab} B$ | holds by assumption (I-S1) |
| S believes fresh(A $\xrightarrow{kab} B$) | holds by assumption (I-S2) |

Derive A's Beliefs after Message 2

Assume:
\textit{Init}
A receives \{na, A $\xrightarrow{kab} B$, fresh(A $\xrightarrow{kab} B$), T\}_kas

Show:
A believes A $\xrightarrow{kab} B$
A believes fresh(A $\xrightarrow{kab} B$)

| A believes S said (na, A $\xrightarrow{kab} B$, fresh(A $\xrightarrow{kab} B$), T) | (R2-1) |
| A believes fresh(na, A $\xrightarrow{kab} B$, fresh(A $\xrightarrow{kab} B$), T) | (R2-2) |

Initial Assumptions

| S believes A $\xrightarrow{kab} B$ | (I-S1) |
| S believes fresh(A $\xrightarrow{kab} B$) | (I-S2) |

| A believes A $\xrightarrow{kab} S$ | (I-A1) |
| A believes S controls A $\xrightarrow{kab} B$ | (I-A2) |
| A believes S controls fresh(A $\xrightarrow{kab} B$) | (I-A3) |
| A believes fresh(na) | (I-A4) |

| B believes B $\xrightarrow{kbs} S$ | (I-B1) |
| B believes S controls A $\xrightarrow{kab} B$ | (I-B2) |
| B believes S controls fresh(A $\xrightarrow{kab} B$) | (I-B3) |
Derive A's Beliefs after Message 2 (cont.)

\[ A \text{ believes } S \text{ believes } A \xrightarrow{kab} B \]  
(R2-4)  
by (R2-3) and rule (BB-Select)

\[ A \text{ believes } S \text{ believes fresh}(A \xrightarrow{kab} B) \]  
(R2-5)  
by (R2-3) and rule (BB-Select)

\[ A \text{ believes } A \xrightarrow{kab} B \]  
(R2-6)  
by (R2-4), (I-A2) and rule (Jurisdiction)

\[ A \text{ believes fresh}(A \xrightarrow{kab} B) \]  
(R2-7)  
by (R2-5), (I-A3) and rule (Jurisdiction)

Show that the Idealization of Message 3 is Sound

There is nothing to show, because message 3 is a ticket: agent \( A \) does not encrypt any part of this message, but only forwards it.

Derive B's Beliefs after Message 3

Assume:

Init

\[ B \text{ receives } \{ A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B) \}_{kbs} \]  
(R3)

Show:

\[ B \text{ believes } A \xrightarrow{kab} B \]  
\[ B \text{ believes fresh}(A \xrightarrow{kab} B) \]  
(R3-1)  
by (R3), (I-B1) and rule (Receives)

▶ We’d like to promote (R3-1) to:

\[ B \text{ believes } S \text{ believes } (A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B)) \]

▶ But we can’t, because the message does not contain a proof of freshness.

▶ We are stuck!

An Attack on NSSK

// \( A \) and \( S \) exchange messages 1 and 2 normally.
\[ A \rightarrow I : \{ msg_3, A, kab \}_{kbs} \] // \( I \) saves message 3.
\[ I \rightarrow B : \{ msg_3, A, kab \}_{kbs} \]
// \( B \) and \( A \) exchange messages 4 and 5 normally.

// Time passes and \( I \) manages to crack \( kab \).
\[ I \rightarrow B : \{ msg_4, nb' \}_{kab} \] // \( I \) sends cracked key to \( B \).
\[ B \text{ generates nonce } nb' \]
\[ B \rightarrow I : \{ msg_4, nb' \}_{kab} \] // \( I \) completes run with \( B \).
\[ I \rightarrow B : \{ msg_5, nb' \}_{kab} \]

Now \( B \) thinks he shares \( kab \) with \( A \).  
But he really shares it with \( I \). Ooops!
Let's add the following unsound assumption to the initial assumptions in order to be able to proceed and see if there are any other bugs in the protocol:

\[ B \text{ believes fresh}(A \xrightarrow{kab} B) \]  
(I-Unsound)

\[ B \text{ believes fresh}(A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B)) \]  
(R3-2)

by (I-Unsound) and rule (Fresh Inject)

\[ B \text{ believes } S \text{ believes } (A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B)) \]  
(R3-4)

by (R3-1), (R3-2) and rule (Fresh)

Form here on, we can derive \( B \text{ believes } A \xrightarrow{kab} B \) and \( B \text{ believes fresh}(A \xrightarrow{kab} B) \) in essential the same way as in

the derivation of \( A \text{ believes } A \xrightarrow{kab} B \) and

\( A \text{ believes fresh}(A \xrightarrow{kab} B) \) above.

## Derive A's Beliefs after Message 4

Assume:

\begin{align*}
\text{Init} \\
A \text{ receives } \{na, A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B), T\}_k\text{as} \\
A \text{ receives } \{A \xrightarrow{kab} B\}_k\text{ab}
\end{align*}

Show:

\begin{align*}
A \text{ believes } B \text{ believes } A \xrightarrow{kab} B \\
A \text{ believes } B \text{ said } A \xrightarrow{kab} B \\
\text{by (R4), (R2-6) and rule (Receives)}
\end{align*}

\begin{align*}
A \text{ believes } B \text{ believes } A \xrightarrow{kab} B \\
\text{by (R4-1), (R2-7) and rule (Fresh)}
\end{align*}

## Derive B's Beliefs after Message 5

The derivation is similar to the derivation of \( A \text{’s beliefs after message 4}. \)

Show that the Idealization of Message 4 is Sound

Assume:

\begin{align*}
\text{Init} \\
B \text{ receives } \{A \xrightarrow{kab} B, \text{fresh}(A \xrightarrow{kab} B)\}_k\text{bs}
\end{align*}

Show:

\( B \text{ believes } A \xrightarrow{kab} B \)

We have shown this already.
Summary of the Analysis

A’s guarantees:
- A believes $A \xrightarrow{kab} B$
- A believes B believes $A \xrightarrow{kab} B$

B’s guarantees under an unsound assumption:
- B believes $A \xrightarrow{kab} B$
- B believes A believes $A \xrightarrow{kab} B$
- The unsound assumption is: B believes fresh($A \xrightarrow{kab} B$)

Homework:
- Fix the protocol using nonces (not timestamps) so that all these guarantees are achieved with no unsound assumptions.
- This will be part of assignment 6.

Further Observations

An inspection of the BAN proof allows us to possibly optimize the protocol by checking what was actually needed in the proof. We make the following observations:
- The encrypted nonce $nb$ in messages 4 and 5 is irrelevant for the agent beliefs. The same beliefs are obtained if, in these messages, $kab$ encrypts any other value.
- The double encryption in message 2 is irrelevant for the agent beliefs. The same beliefs would be obtained if message 2 were replaced by the following:
  \begin{itemize}
  \item 2. $S \rightarrow A : \{msg_2, na, B, kab\}_{kas}, \{msg_3, kab, A\}_{kbs}$
  \end{itemize}

Homework

- I have posted a homework on BAN logic. This is due next week, on April 7, in class.

A Fix Using Timestamps: Denning–Sacco

\begin{itemize}
  \item $A \rightarrow S : A, B$
  \item $S$ reads the clock and binds the current time to $t$
  \item $S \rightarrow A : \{t, B, kab, \{t, A, kab\}_{kbs}\}_{kas}$
  \item $A$ checks if $t$ is close to the current time
  \item $A \rightarrow B : \{t, A, kab\}_{kas}$
  \item $B$ checks if $t$ is close to the current time
\end{itemize}

Caveat:
- Timestamp-based protocols require synchronized local clocks.
- Synchronizing local clocks over an untrusted network is itself a security problem.
M. Burrows, M. Abadi, and R.M. Needham.  
A logic of authentication. 

P. Syverson and I. Cervesato.  
The logic of authentication protocols. 