

A tool for an automatic analysis of security protocols by logics of belief

Marián Novotný

Institute of Computer Science
P.J. Šafárik University, Faculty of Science
Košice, Slovakia

MEMICS 2008 - Annual Doctoral Workshop on
Mathematical and Engineering Methods in Computer
Science

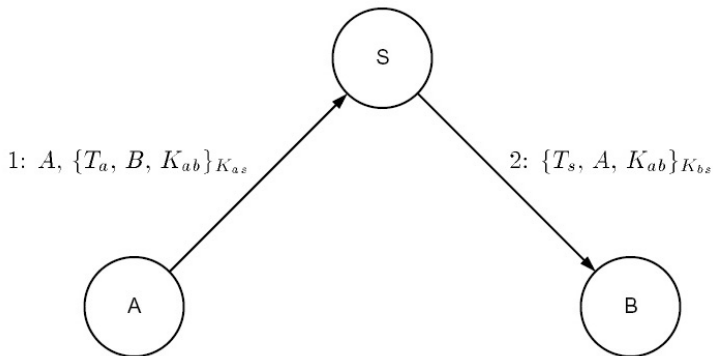
Introduction - Cryptographic protocols

- very important mechanisms for **securing** computer systems
- use **cryptographic primitives** such as (a)symmetric encryption, digital signatures, hash functions, nonces, timestamps etc.
- The most significant classes are protocols for
 - **authentication** - ensuring the identity of the participant
 - **key establishment** - creating and distributing secret keys

Example of a key establishment protocol - "Wide-mouthed frog"

(1) $A \longrightarrow S : A, \{T_A, B, K_{AB}\}_{K_{AS}}$

(2) $S \longrightarrow B : \{T_S, A, K_{AB}\}_{K_{BS}}$



Formal analysis of cryptographic protocols

- based on **Delov-Yao** model of an attacker, assumes
 - **perfect** cryptography
 - network is under complete **control** of an attacker
- various approaches
 - **logics of belief** - Burrows, Abadi, Needham -BAN, AUTLOG, GNY - SPEAR
 - **model checking** - Lowe - Casper (CSP-FDR)
 - **inductive approach** - Paulson - Isabelle
 - **strand spaces** - Herzog, Guttman - Athena (Song), Scyther (Cremers)
 - **the applied pi-calculus** - Abadi - ProVerif (Blanchet)
 - The **AVISPA** project

Analysis of security protocols by logics of belief

- Questions for which the designers are trying to find answers:
 - What does the protocol **achieve**?
 - Which **assumptions** the protocol needs?
 - Does the protocol something **unnecessary**, which can be left out?
- The process of analysis of protocol consists of four phases:
 - The **idealization** of the protocol - the translation into the formalism of the BAN logic.
 - Formulating **assumptions** of the protocol - the assignment of initial beliefs of principals.
 - Formulating **goals** of the protocol.
 - Deciding whether goals can be **derived** and which assumptions are used by the derivation.

Some function symbols of the BAN logic

- $P \models X$ P **believes** X . The principal P may act as though the formula X is true.
- $P \triangleleft X$ P **sees** X . Someone has sent a message containing X to P .
- $P \sim X$ P **once said** X . P at some time sent a message including the statement X .
- $P \models X$ P has **jurisdiction** over X . P is an authority on X and should be trusted.
- $\sharp(X)$ A formula X is **fresh** and has not been sent in a message before.
- $P \stackrel{K}{\leftrightarrow} Q$ K is a **shared key** for principals P, Q .
The key K is good, it is not compromised.
- $\{X\}_K$ A formula X is **encrypted** under a key K .
- (X, Y) The **concatenation** of formulae X, Y .

Some inference rules of the BAN logic

$$1. \frac{P \triangleleft \{X\}_K, P \models P \xleftarrow{K} Q}{P \models Q \sim X}$$

If P **sees** a formula X encrypted under a key K and P **believes** that K is shared with Q , then P **believes** that Q once said X .

$$2. \frac{P \models \#(X)}{P \models \#((X, Y))}$$

If one part of a formula is known to be **fresh**, then the entire formula also must be **fresh**.

$$3. \frac{P \models Q \sim X, P \models \#(X)}{P \models Q \models X}$$

If P **believes** that Q **once said** X and **believes** X is **fresh** then P **believes** that Q **believes** X .

$$4. \frac{P \models Q \models (X, Y)}{P \models Q \models Y}$$

If a participant **believes** a statement, then he **believes** also each part.

$$5. \frac{P \models Q \Rightarrow X, P \models Q \models X}{P \models X}$$

If P **believes** that Q has **jurisdiction** over X and P **believes** that Q **believes** X , then P **believes** X .

Derivation in BAN logic in "Wide-mouthed frog" protocol

- 1 $B \models B \xleftrightarrow{K_{BS}} S$ the **assumption**
- 2 $B \triangleleft \left\{ (T_S, A \models A \xleftrightarrow{K_{AB}} B) \right\}_{K_{BS}}$ the **idealized** second message
- 3 $B \models \#(T_S)$ the **assumption**
- 4 $B \models S \Rightarrow A \models A \xleftrightarrow{K_{AB}} B$ the **assumption**
- 5 $B \models A \Rightarrow A \xleftrightarrow{K_{AB}} B$ the **assumption**
- 6 $B \models S \sim (T_S, A \models A \xleftrightarrow{K_{AB}} B)$ **application** of the rule (1) to 2,1
- 7 $B \models \#((T_S, A \models A \xleftrightarrow{K_{AB}} B))$ **application** of the rule (2) to 3
- 8 $B \models S \models (T_S, A \models A \xleftrightarrow{K_{AB}} B)$ **application** of the rule (3) to 6,7
- 9 $B \models S \models A \models A \xleftrightarrow{K_{AB}} B$ **application** of the rule (4) to 8
- 10 $B \models A \models A \xleftrightarrow{K_{AB}} B$ **application** of the rule (5) to 4,9
- 11 $B \models A \xleftrightarrow{K_{AB}} B$ **application** of the rule (5) to 5,10

Decision procedure for an automatic analysis

- should **decide** whether a formula can be derived from a set of assumptions
- There are two basic strategies for derivation:
 - The **forward** derivation starts with a set of assumptions and in each round extends it by all derivable terms.
 - The **backward** derivation starts with a goal and finds all rules which conclusions can be unified with the goal. After applying the unification to premises of the rule it continues recursively for all these subgoals.
- Unfortunately we can not **directly apply** these methods
 - $\frac{P \models \#(X)}{P \models \#((X, Y))}$ is not suitable for the forward search
 - $\frac{P \triangleleft \{X\}_K, P \models P \xrightarrow{K} Q}{P \models Q \sim X}$ for the backward derivation
- We divide rules of a logic into two disjoint sets:
 - **compositions rules** - suitable for backward derivation
 - **decompositions rules** - suitable for forward derivation. In these rules we distinguish between **main** and **auxiliary** premisses

Transformation of logics

- for each function symbol except constants we add the **special symbol** \square to the set of function symbols
- the set of function symbols of transformed logic $FS' = FS \cup \{\langle \square f, i \rangle; \langle f, i \rangle \in FS \wedge i > 0\}$.
- for the set of rules \vdash we define the set of transf. rules \vdash'
 - if the rule $\frac{\mathcal{H}_1, \dots, \mathcal{H}_n}{\mathcal{C}}$ is a **composition**
 - 1 $\frac{\square \mathcal{C}, \mathcal{H}_1, \dots, \mathcal{H}_n}{\mathcal{C}}$
 - 2 $\frac{\square \mathcal{C}}{\square \mathcal{H}_i}$ for each $i \in \{1, \dots, n\}$
 - if the rule $\frac{\mathcal{H}_1, \dots, \mathcal{H}_m, \mathcal{A}_1, \dots, \mathcal{A}_k}{\mathcal{C}}$, $m > 0, k \geq 0$ is a **decomposition**
 - 1 $\frac{\mathcal{H}_1, \dots, \mathcal{H}_m, \mathcal{A}_1, \dots, \mathcal{A}_k}{\mathcal{C}}$,
 - 2 if $k \geq 1$, then also $\frac{\mathcal{H}_1, \dots, \mathcal{H}_m}{\square \mathcal{A}_i}$ for each $i \in \{1, \dots, k\}$

Lemma

For a logic $\langle FS, \vdash \rangle$ the set of transformed rules \vdash' is suitable for the forward derivation.

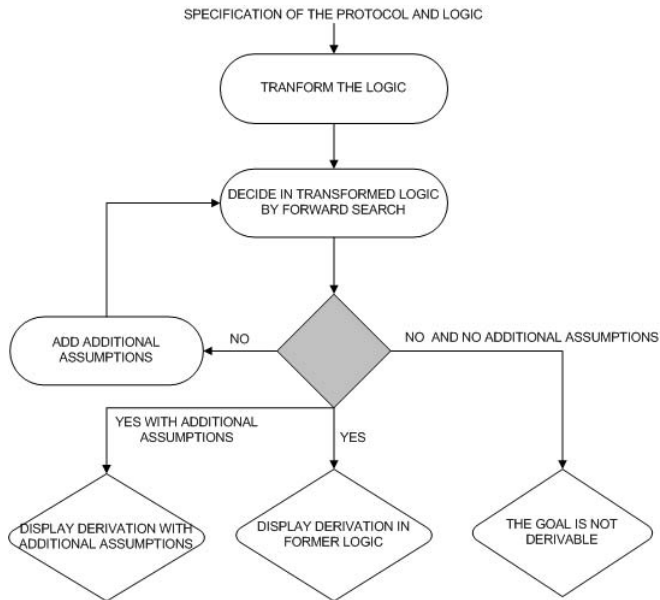
Theorem

For a logic $\langle FS, \vdash \rangle$ which fulfills the normal derivation criterion^a, for a set of assumptions Γ and a goal t , it holds

$$\Gamma \cup \{\Box t\} \vdash' t \equiv \Gamma \vdash t.$$

^aIf we have derived some formula C by applying the decomposition rule $\frac{\mathcal{H}_1, \dots, \mathcal{H}_m, A_1, \dots, A_k}{C}$ to constant terms $H_1, \dots, H_m, A_1, \dots, A_k$ then H_1, \dots, H_m were not derived using a composition rule.

The process of the derivation in the ABLOB program



Editor of logic in ABLOB

```
ABLOB
File Edit Protocol Options Help
BAN Logic | Wide-mouthed-frog Protocol

LOGIC BAN
FUNCTIONS
believes 2
sees 2
said 2
jurisdiction 2
fresh 1
sharedkey 3
publickey 2
secret 3
encrypt 2
combine 2
pair 2
inv 2
RULES
%message meaning
believes(?P,said(?Q,?X)):-sees(?P,encrypt(?X,?K)),believes(?P,sharedkey(?P,?Q,?K))
believes(?P,said(?Q,?X)):-sees(?P,combine(?X,?Y)),believes(?P,secret(?P,?Q,?Y))
believes(?P,said(?Q,?X)):-sees(?P,encrypt(?X,?S)),believes(?P,publickey(?P,?Q)),inv(?V,?S)
%nonce-verification
Abelieves(?P,believes(?Q,?X)):-believes(?P,said(?Q,?X)),believes(?P,fresh(?X))
%believes(?P,believes(?Q,?X)):-sees(?P,combine(?X,?Y)),believes(?P,secret(?P,?Q,?Y)),believ
believes(?P,fresh(?X)):-sees(?P,combine(?X,?Y)),believes(?P,secret(?P,?Q,?Y)),believes(?P,f
%believes(?P,fresh(?X)):-sees(?P,encrypt(?X,?K)),believes(?P,sharedkey(?P,?Q,?K)),believes(?P,
%jurisdiction
believes(?P,?X):-believes(?P,jurisdiction(?Q,?X)),believes(?P,believes(?Q,?X))
%symmetric keys
Abelieves(?P,believes(?Q,sharedkey(?R,?S,?K))):-believes(?P,believes(?Q,sharedkey(?S,?R,?K)))
Abelieves(?P,believes(?Q,secret(?R,?S,?Y))):-believes(?P,believes(?Q,secret(?S,?R,?Y)))
Abelieves(?P,secret(?Q,?R,?Y)):-believes(?P,secret(?R,?Q,?Y))
Abelieves(?P,sharedkey(?Q,?R,?K)):-believes(?P,sharedkey(?R,?Q,?K))
%sees
Asees(?P,pair(?X,?Y)):-sees(?P,pair(?Y,?X))
sees(?P,?X):-sees(?P,pair(?X,?Y))
sees(?P,?X):-sees(?P,encrypt(?X,?K)),believes(?P,sharedkey(?P,?Q,?K))
sees(?P,?X):-sees(?P,encrypt(?X,?K)),believes(?P,publickey(?K,?P))
sees(?P,?X):-sees(?P,encrypt(?X,?S)),believes(?P,publickey(?V,?Q)),inv(?V,?S)
sees(?P,?X):-sees(?P,combine(?X,?Y))
```

Editor of protocol in ABLOB

```
ABLOB
File Edit Protocol Options Help
BAN Logic Wide-mouthed-Frog Protocol
PROTOCOL wide-mouthed-Frog
LOGIC BAN
CONSTANTS
%Principals
A
B
S
%Keys
K_{AS}
K_{AB}
K_{BS}
%Timestamps
T_S
T_A
ASSUMPTIONS
%Messages
%1. A->S: A, {T_A, B, K_AB}_{K_AS}
sees(S, encrypt(pair(T_A, sharedkey(A, B, K_{AB})), K_{AS}))
%2. S->B: {T_S, A, K_AB}_{K_BS}
sees(B, encrypt(pair(T_S, believes(A, sharedkey(A, B, K_{AB}))), K_{BS}))
%initial believes
believes(A, sharedkey(A, S, K_{AS}))
believes(B, sharedkey(B, S, K_{BS}))
believes(S, sharedkey(A, S, K_{AS}))
believes(S, sharedkey(B, S, K_{BS}))
believes(A, sharedkey(A, B, K_{AB}))
believes(S, fresh(T_A))
believes(B, fresh(T_S))
believes(B, jurisdiction(A, sharedkey(A, B, K_{AB})))
believes(B, jurisdiction(S, believes(A, sharedkey(A, B, K_{AB}))))
GOALS
believes(B, sharedkey(A, B, K_{AB}))
believes(B, believes(A, sharedkey(A, B, K_{AB})))
```

Derivation results in ABLOB

ABLOB

File Edit Protocol Options Help

Analysis of The Wide-mouthed-frog Protocol

Analysis of The Wide-mouthed-frog Protocol
Logic: BAN

Assumptions:

```
1 believes(A, sharedkey(A, B, K_{AB}))
2 believes(A, sharedkey(A, S, K_{AS}))
3 believes(B, jurisdiction(A, sharedkey(A, B, K_{AB})))
4 believes(B, jurisdiction(S, believes(A, sharedkey(A, B, K_{AB}))))
5 believes(B, fresh(T_S))
6 believes(B, sharedkey(B, S, K_{BS}))
7 believes(S, fresh(T_A))
8 believes(S, sharedkey(A, S, K_{AS}))
9 believes(S, sharedkey(B, S, K_{BS}))
10 sees(B, encrypt(pair(T_S, believes(A, sharedkey(A, B, K_{AB}))), K_{BS}))
11 sees(S, encrypt(pair(T_A, sharedkey(A, B, K_{AB}))), K_{AS}))
```

Goals:

```
A believes(B, sharedkey(A, B, K_{AB}))
```

Derivation:

```
1 believes(B, sharedkey(B, S, K_{BS})) assumption
2 sees(B, encrypt(pair(T_S, believes(A, sharedkey(A, B, K_{AB}))), K_{BS})) assumption
3 believes(B, fresh(T_S)) assumption
4 believes(B, jurisdiction(S, believes(A, sharedkey(A, B, K_{AB})))) assumption
5 believes(B, jurisdiction(A, sharedkey(A, B, K_{AB}))) assumption
6 believes(B, said(S, pair(T_S, believes(A, sharedkey(A, B, K_{AB})))) applying rule believes(P, said(Q, X))
7 believes(B, fresh(pair(T_S, believes(A, sharedkey(A, B, K_{AB})))) applying rule believes(P, fresh(Q, X))
8 believes(B, believes(S, pair(T_S, believes(A, sharedkey(A, B, K_{AB})))) applying rule believes(P, believes(Q, X))
9 believes(B, believes(S, believes(A, sharedkey(A, B, K_{AB})))) applying rule believes(P, believes(Q, Y))
10 believes(B, believes(A, sharedkey(A, B, K_{AB}))) applying rule believes(P, X):-believes(P, jurisdiction(Q, X)),believes(B, sharedkey(A, B, K_{AB}))
11 believes(B, sharedkey(A, B, K_{AB})) applying rule believes(P, X):-believes(P, jurisdiction(Q, X)),believes(B, sharedkey(A, B, K_{AB}))
```

Undesirable Assumptions:

1 Analysis of The Wide-mouthed-frog Protocol

- Logic: BAN

1.1 Assumptions

1. $A \models A \stackrel{K_{AB}}{\longleftrightarrow} B$
2. $A \models A \stackrel{K_{AS}}{\longleftrightarrow} S$
3. $B \models A \mapsto A \stackrel{K_{AB}}{\longleftrightarrow} B$
4. $B \models S \mapsto A \models A \stackrel{K_{AB}}{\longleftrightarrow} B$
5. $B \models \#(T_S)$
6. $B \models B \stackrel{K_{BS}}{\longleftrightarrow} S$
7. $S \models \#(T_A)$
8. $S \models A \stackrel{K_{AS}}{\longleftrightarrow} S$
9. $S \models B \stackrel{K_{BS}}{\longleftrightarrow} S$
10. $B \triangleleft \{(T_S, A \models A \stackrel{K_{AB}}{\longleftrightarrow} B)\}_{K_{BS}}$
11. $S \triangleleft \{(T_A, A \stackrel{K_{AS}}{\longleftrightarrow} B)\}_{K_{AS}}$

1.2 Goals

I $B \models A \stackrel{K_{AB}}{\longleftrightarrow} B$

Derivation:

- 1 $B \models B \stackrel{K_{BS}}{\longleftrightarrow} S$ assumption
- 2 $B \triangleleft \{(T_S, A \models A \stackrel{K_{AB}}{\longleftrightarrow} B)\}_{K_{BS}}$ assumption
- 3 $B \models \#(T_S)$ assumption
- 4 $B \models S \mapsto A \models A \stackrel{K_{AB}}{\longleftrightarrow} B$ assumption
- 5 $B \models A \mapsto A \stackrel{K_{AB}}{\longleftrightarrow} B$ assumption

Implementation of the BAN logic in ABLOB

- **function symbols** are defined following the paper [BAN89]
- for the exact relation between public and private keys we extended this set by the binary function $inv(P, S)$
- we imported all rules from the paper [BAN89], because of **symmetry of concatenation** we appended symmetric versions of all rules
- we **divided** all rules into **composition** and **decomposition** classes according to the normal derivation criterion
- we **analyzed** all protocols from the paper [BAN89] and appended some **missed** assumptions and rules

Implementation of the AUTLOG logic in ABLOB

- the set of function symbols is **extension** of the BAN logic
- because of the **restriction** of the decision procedure we can not directly use rules such as
$$\frac{P \triangleleft P \xleftrightarrow{K} Q, P \models \#(P \xleftrightarrow{K} Q)}{P \models \#(\{X\}_K)}$$
- instead of this rule we added an **auxiliary function symbol** $\alpha(P, K)$ and one composition and decomposition rule
$$\frac{P \triangleleft P \xleftrightarrow{K} Q, P \models \#(P \xleftrightarrow{K} Q)}{\alpha(P, K)}, \frac{\alpha(P, K)}{P \models \#(\{X\}_K)}$$
- in this way we modified seven rules with respect to the normal derivation criterion.
- we **analyzed** simple challenge-response protocols from the paper [AUTLOG94]
- we also **compared** AUTLOG with BAN by the analysis of the Kerberos protocol

Conclusions

- we designed and implemented the **user-friendly tool** for an automatic analysis of cryptographic protocols by logics of belief
- the derivation is based on the **combination** of forward and backward search
- in this tool we **implemented** BAN and AUTLOG logics and analyzed some protocols from literature
- formal analysis by logics of belief is not under current research in this field, but we hope that our user-friendly tool with the database of protocols from literature can be useful for **teaching** cryptography, especially cryptographic protocols

Thank you for your attention

Thank you for your attention