Formal Methods for Assuring Security of Computer Networks

Siavash Soleimanifard

May 8, 2012
Outline

1. Testing
2. Formal Methods
   - Tools for formal methods
   - Model–based software development
3. Security
   - Principals of security
   - Key security properties
   - Assessing security protocols
   - Needham–Schroeder public–key protocol
4. BAN logic
   - Analyzing Needham–Schroeder by BAN logic
5. MCA
   - MCA
6. Conclusion
Goals

- Introduction to formal methods
- Motivation of using formal methods for assuring security
- Comparing with testing techniques
- We use Needham-Schroeder public-key protocol
  - Logical method
  - Modelling the messages
  - Specifying and assessing authentication
Testing

- Important part of producing each and every product
- More important in IT area, More complicated
- More than 50% of software development process
- Software bugs cost the U.S. economy $59.5 billion annually [NIST report, 2006]
What is Formal Methods?

- **Formal Methods Definition**
  - Mathematical based techniques for the specification, development and verification of software and hardware systems.

- **Why do we use formal methods?**
  - To prove that the system behaves as it is expected

- **How do we use it?**
  - Model the systems mathematically
  - Prove that the system satisfy the properties we are interested in

- **Why don’t we always use formal method techniques**
  - Formal method techniques are expensive to apply
Tools for formal methods

- Automata
- Logic and Calculi
- Formal languages
- Semantics
- Algebra (e.g., process algebra)
Example of using formal methods

Formal Methods

Conclusion

BAN logic

Security

MCA

Model-based software development

Formal Methods for Assuring Security of Computer Networks

Siavash Soleimanifard

Formal Methods
Model–based software development

- Design a model
- Verify the model
- Implement the model
- Validate the implementation
- Take existing implementation
- Extract a model from implementation
- Verify the model
Different modelling techniques

- Models are extracted from the code
- Models are extracted from the requirements and specification
- Models are extracted from a black-box
Security is achieved by:

- Sufficiently strong cryptographic algorithms and protocols
- Correct implementation of hardware and software
- Appropriate assumptions about trusted authorities
- Formal analytical techniques
Principals of security

- People
- Keys
- Processes and machines that send and receive information
- Access information resources
Key security properties

- **privacy or confidentiality**: knowing with accuracy which principals can read data
- **integrity**: detecting corruption of information
- **authentication**: knowing the identity of a principal or the source of information
- **access control**: restricting the use of a resource to privileged principals
- **non-repudiation**: preventing principals from subsequently denying their actions
- **availability of service**: guaranteeing authorised principals always have access to services
Public–key infrastructure

- Encryption through pairs of keys
  - Public which is made known to others
  - Private which should be known only to the principal
- Supports both privacy and digital signatures

Figure:

Sign (Encrypt)

DFCD3454
BBEA788A

Alice's private key

Verify (Decrypt)

Alice's public key

Alice

I will pay $500

Bob

I will pay $500

Hello Alice!

Encrypt

6EB69570
08E03CE4

Alice's public key

Decrypt

Hello Alice!

Sign (Encrypt)

Alice

Verify (Decrypt)

Bob

Alice's private key

Alice's public key

Siavash Soleimanifard

Formal Methods for Assuring Security of Computer Networks
Defeat a protocol by using bits of previously successful protocol runs (known as replay attacks)

or by some form of impersonation (e.g., man in the middle attacks)
key–distribution protocols that enable secure communication sessions where both parties are active at the same time

The Needham-Schroeder protocol is designed to allow two principals to mutually authenticate themselves through a series of message exchanges, as a prelude for some secure session
FIGURE 1. Needham-Schroeder protocol

1. $A \rightarrow S : A, B$
2. $S \rightarrow A : \{K_b, B\}_{K_S^{-1}}$
3. $A \rightarrow B : \{N_a, A\}_{K_b}$
4. $B \rightarrow S : B, A$
5. $S \rightarrow B : \{K_a, A\}_{K_S^{-1}}$
6. $B \rightarrow A : \{N_a, N_b\}_{K_a}$
7. $A \rightarrow B : \{N_b\}_{K_b}$
BAN logic

- \( P \models X \), read as “Principal \( P \) believes \( X \)”. \( P \) behaves as if \( X \) is true.
- \( P \triangleleft X \), read as “\( P \) sees \( X \)”. A principal has sent \( P \) a message containing \( X \).
- \( P \vdash X \), read as “\( P \) once said \( X \)”. \( P \) at some time (either the present or the past) believed \( X \) and sent it as part of a message.
- \( P \Rightarrow X \), read as “\( P \) has jurisdiction over \( X \)”. Principal \( P \) has authority over \( X \) and is trusted on this matter.
- \( X(X) \), read as “\( X \) is fresh”. \( X \) has not appeared in a past run of the protocol.
- \( K \rightarrow P \), read as “\( P \) has public key \( K \)”. The corresponding private key is denoted by \( K^{-1} \) and assumed to be known only by \( P \).
Analyzing Needham–Schroeder by BAN logic

Inference Rules

\[
\begin{align*}
P & \models \Rightarrow Q & P & \triangleleft \{X\}_{K-1} & \quad \text{Message-Meaning Rule for Public-Keys} \\
& & P & \models (Q \Rightarrow X) & \\
\hline
P & \models \mathcal{V}(X) & P & \models (Q \Rightarrow X) & \quad \text{Nonce-Verification Rule} \\
& & P & \models (Q \models X) & \\
\hline
P & \models (Q \models X) & P & \models (Q \models X) & \quad \text{Jurisdiction Rule} \\
& & P & \models X & 
\end{align*}
\]
Analyzing Needham–Schroeder by BAN logic

FIGURE 1. Needham-Schroeder protocol

1. $A \to S : A, B$
2. $S \to A : \{K_b, B\}_{K_s^{-1}}$
3. $A \to B : \{N_a, A\}_{K_b}$
4. $B \to S : B, A$
5. $S \to B : \{K_a, A\}_{K_s^{-1}}$
6. $B \to A : \{N_a, N_b\}_{K_a}$
7. $A \to B : \{N_b\}_{K_b}$
After step 2,

\[ A \models (\frac{K_b}{\rightarrow} B) \]

is the desired property.
After step 2,

\[ A \models (\frac{K_b}{\Rightarrow} B) \]

is the desired property.

Live proof
- $CA_1 \langle CA_2 \rangle$
- $CA_2 \langle B \rangle$
\[ CA_2 p = CA_1 p \quad \bigcirc \quad CA_1 \langle CA_2 \rangle \]

\[ B_p = CA_2 p \quad \bigcirc \quad CA_2 \langle B \rangle \]
Notions and Axioms

Notions

- $P$ says $s$
- $P \Rightarrow Q$
Notions and Axioms

**Notions**
- $P$ says $s$
- $P \Rightarrow Q$

**Axioms**
- $(P$ says $(Q \Rightarrow P)) \supset (Q \Rightarrow P)$
- $(P \Rightarrow Q) \supset ((P$ says $s) \supset (Q$ says $s))$
1. \[ CA_1 \Rightarrow CA1 \]
   A knows the public key of its certification authority (i.e., from A’s perspective, the key \( CA_1 \) speaks for \( CA1 \)).

2. \[ CA_1 \text{ says } (CA_2 \Rightarrow CA2) \]
   \( CA1 \)'s public key \( CA_1 \) is used to verify the validity of the certificate \( CA1 \langle CA2 \rangle \).

3. \[ CA_2 \text{ says } (B \Rightarrow B) \]
   Similarly, the key \( CA_2 \) is used to verify the certificate \( CA2 \langle B \rangle \).

4. \[ CA1 \Rightarrow CA2 \]
   A trusts its certificate authority to speak for other certificate authorities (\( CA2 \) in this case).

5. \[ CA2 \Rightarrow B \]
   A knows that B’s certificate authority is \( CA2 \).
The desired property is that $A$ concludes $B_p \Rightarrow B$. 
The desired property is that A concludes $B_p \Rightarrow B$.

Live proof
Formal methods is an interesting method for proving the correctness of programs, e.g., security protocols. It is costly but worth.
References


Thank You For Listening