Security Protocols and Their Correctness

Lawrence C. Paulson

Computer Laboratory

University of Cambridge
Can Cryptography Make Networks Secure?

Goals:

- **Authenticity**: who sent it?
- **Secrecy**: who can receive it?

Threats:

- **Active** attacker
- **Careless & compromised agents** … **NO code-breaking**
Some Notation

\[ A, B \] \quad \text{agent names} \ (\text{Alice, Bob})

\[ Na \] \quad \text{nonce chosen by Alice} \ (\text{a random number})

\[ Ka \] \quad \text{Alice’s public key}

\[ \{ X \}_{Ka} \] \quad \text{message encrypted using} \ Ka

- \text{anybody} can encrypt
- \text{only Alice} can recover \ X
The Needham-Schroeder Protocol

1. \[ A \rightarrow B : \{ Na, A \}^K_b \]
   
   Alice sends Bob an encrypted nonce

2. \[ B \rightarrow A : \{ Na, Nb \}^K_a \]
   
   Bob returns \( Na \) with a nonce of his own

3. \[ A \rightarrow B : \{ Nb \}^K_b \]
   
   Alice returns Bob’s nonce


What Does Needham-Schroeder Accomplish?

Only Bob could recover $Na$

Only Alice could recover $Nb$

- Therefore Alice and Bob are present now

But are the nonces secret?
A Middle-Person Attack

Villain Charlie can masquerade as Alice to Bob

\[ \{A, Na\}K_c \quad \{A, Na\}K_b \]
\[ \{Nb\}K_c \quad \{Nb\}K_b \]
Lowe’s Attack in Detail

1. \( A \rightarrow C : \{Na, A\}_{Kc} \)
1’. \( C(A) \rightarrow B : \{Na, A\}_{Kb} \)

2. \( B \rightarrow C(A) : \{Na, Nb\}_{Ka} \)

2. \( C \rightarrow A : \{Na, Nb\}_{Ka} \)

3. \( A \rightarrow C : \{Nb\}_{Kc} \)

3’. \( C(A) \rightarrow B : \{Nb\}_{Kb} \)

Can protocols be verified?
Verification Method I: Authentication Logics

**BAN logic**: Burrows, Abadi, Needham (1989)

Models agent **beliefs**:

- Nonce $\mathcal{N}$ is fresh  
- Key $K_{ab}$ is good

Agent S can be trusted

- Allows short, abstract proofs but misses many flaws
Verification Method II: State Enumeration

Specialized tools (Meadows, Millen)

General model-checkers (Lowe)

Model protocol as a finite-state system

- Automatically finds attacks but requires strong assumptions

Can we use formal proof?
Inductive Protocol Verification

- Traces of events: $A$ sends $X$ to $B$
- Operational model of agents
- Algebraic theory of messages (derived)
- A general attacker
- Proofs mechanized using Isabelle/HOL
Sets of Messages

parts $H$: the components of $H$

$\text{Crypt } KX \leftrightarrow X$

analz $H$: the accessible components of $H$

$\text{Crypt } KX, K^{-1} \leftrightarrow X$

synth $H$: messages that can be made from $H$

$X, K \leftrightarrow \text{Crypt } KX$

Defined inductively
Some Algebraic Laws

\[
\begin{align*}
\text{parts}(\text{parts } H) &= \text{parts } H \\
\text{parts}(\text{analz } H) &= \text{parts } H \\
\text{analz}(\text{synth } H) &= \text{analz } H \cup \text{synth } H \\
\text{synth}(\text{analz } H) &= ??
\end{align*}
\]

Keep the 3 notions separate

Model as set transformers
If a trace has the event

\[
\text{Says } A' \ B \ (\text{Crypt(pubK } B\{Na, A\})
\]

and \(Nb\) is fresh, then may add the event

\[
\text{Says } B \ A \ (\text{Crypt(pubK } A\{Na, Nb\})
\]

\(B\) doesn’t know the true sender (shown as \(A'\))
Modelling Attacks and Accidents

Fake. If $X \in \text{synth}(\text{analz}(\text{spies } evs))$

may add the event

Says Spy $B X$

Can also model accidents: giving secrets away

Does one compromise lead to others?
Facts that Can be Proved

- Secret keys are never lost
- Nonces uniquely identify their message of origin
- Nonces stay secret (under certain conditions!)

Proved by induction, simplification & classical reasoning

Simplification of analz: case analysis, big formulas
Final Remarks

- A dozen protocols analyzed:
  (Otway-Rees, Yahalom, Needham-Schroeder, . . .)

- **TLS**: an Internet protocol

- 2–9 minutes **CPU time** per protocol

- few hours or days **human time** per protocol

- a good **complement** to model-checking