Mechanized Proofs for a Recursive Authentication Protocol

Lawrence C. Paulson

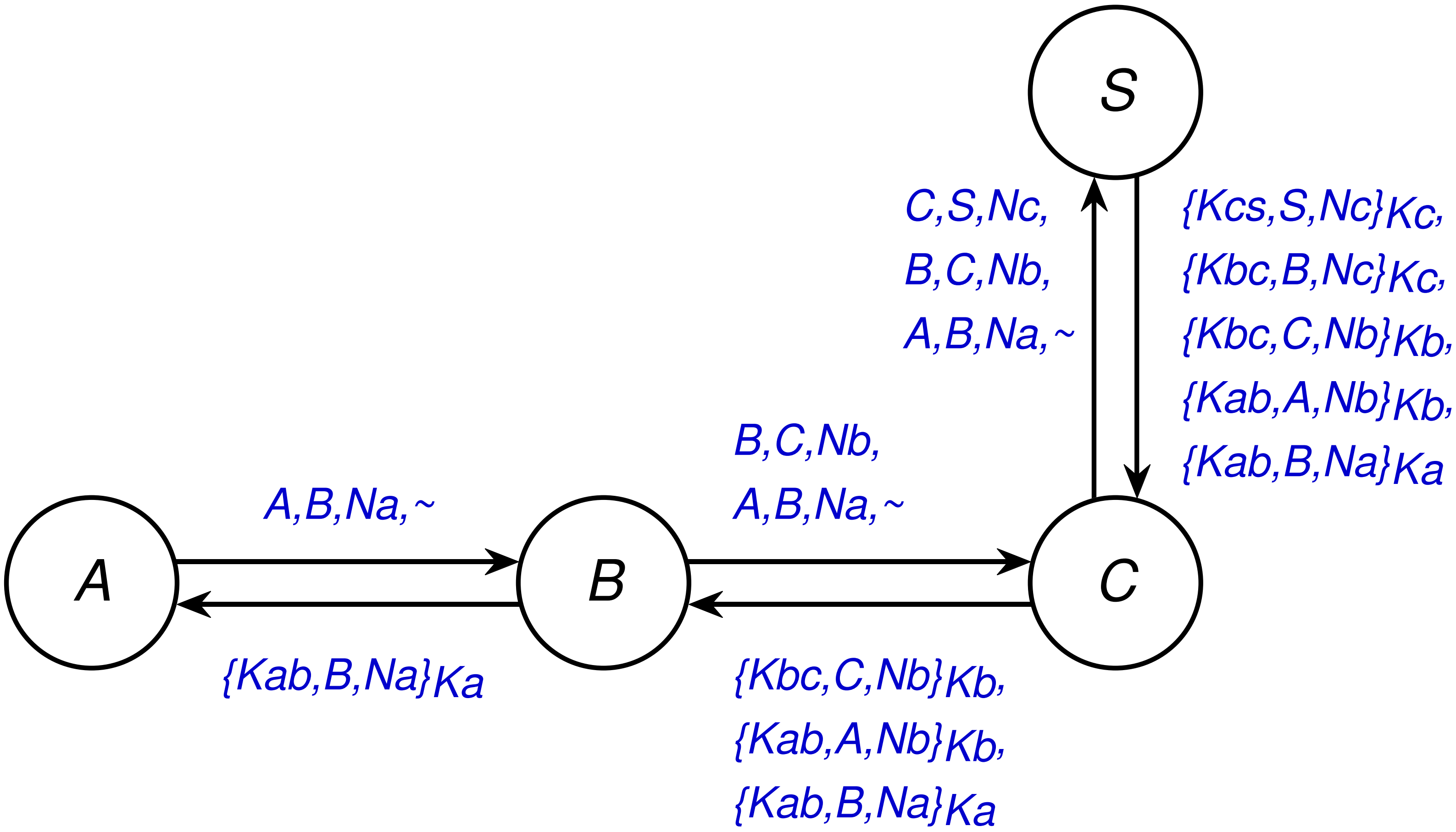
Computer Laboratory

University of Cambridge

Overview of the Protocol

- Based on Otway-Rees
- Distributes session keys for any number of agents
- Can be implemented as remote procedure calls
- “application components are in control of security policy and its enforcement” — John Bull
- Some modifications to assist proofs

The Protocol, with 3 Clients



The Protocol: Accumulation of Requests

Hashing to make Message Authentication Codes:

$$\text{Hash}_X\, Y \equiv \{\!|\text{Hash}\{\!|X, Y|\!\}, Y|\!\}$$

$$\begin{array}{lll}
1. & A \to B & : \text{Hash}_{Ka}\{\!|A, B, Na, -|\!\} \\
2. & B \to C & : \text{Hash}_{Kb}\{\!|B, C, Nb, \text{Hash}_{Ka}\{\!|A, B, Na, -|\!\}|\!\} \\
2'. & C \to S & : \text{Hash}_{Kc}\{\!|C, S, Nc, \text{Hash}_{Kb}\{\!|B, C, Nb, \cdots|\!\}|\!\}
\end{array}$$

No limit on the nesting of requests

The Protocol: Distribution of Certificates

$$
\begin{array}{lrl}
3. & S \to C : & \{\!|Kcs, S, Nc|\!\}_{Kc}, \{\!|Kbc, B, Nc|\!\}_{Kc}, \\
 & & \{\!|Kbc, C, Nb|\!\}_{Kb}, \{\!|Kab, A, Nb|\!\}_{Kb}, \\
 & & \{\!|Kab, B, Na|\!\}_{Ka} \\
4. & C \to B : & \{\!|Kbc, C, Nb|\!\}_{Kb}, \{\!|Kab, A, Nb|\!\}_{Kb}, \\
 & & \{\!|Kab, B, Na|\!\}_{Ka} \\
4'. & B \to A : & \{\!|Kab, B, Na|\!\}_{Ka}
\end{array}
$$

The Verification Method

- Formal proof, not finite state checking
- Trace semantics, no beliefs or other modalities
- Inductive definitions: a simple, general model of action
- Any number of interleaved runs
- A general & uniform attacker
- Mechanized using Isabelle/HOL

Processing Message Histories

- parts: message components $\quad$ Crypt $K\,X \leadsto X$

  parts H contains everything potentially recoverable from H

- analz: message decryption $\quad$ Crypt $K\,X,\ K^{-1} \leadsto X$

  analz H contains everything currently recoverable from H

- synth: message faking $\quad$ $X,\ K \leadsto$ Crypt $K\,X$

  synth H contains everything expressible using H

The Introduction of Hashing

Allow the message Hash X. How to extend the operators?

- Don't add Hash $X \in$ parts $H \implies X \in$ parts H
- Don't add Hash $X \in$ analz $H \implies X \in$ analz H
- Do add $X \in$ synth $H \implies$ Hash $X \in$ synth H

Hashing is one-way, so hash values are atomic

Components *vs* Ingredients

Inductively Defining the Protocol, 1–2

1. If evs is a trace and Na is fresh, may add

$$\text{Says}\ A\ B\ (\text{Hash}_{\text{shrK}\ A}\{\!|A, B, Na, -|\!\})$$

2. If evs has Says $A'\ B\ Pa$ and $Pa = \{\!|Xa, A, B, Na, P|\!\}$ and Nb is fresh, may add

$$\text{Says}\ B\ C\ (\text{Hash}_{\text{shrK}\ B}\{\!|B, C, Nb, Pa|\!\})$$

B doesn't know the true sender & can't verify hash Xa

Inductively Defining the Protocol, 3–4

3. If evs contains the event Says B' S Pb, may add a suitable response Says S B Rb

4. If evs contains the events

$$\text{Says}\ B\ C\ (\text{Hash}_{\text{shrK}\ B}\{\!|B, C, Nb, Pa|\!\})$$

$$\begin{aligned}\text{Says}\ C'\ B\ \{\!|\ &\text{Crypt}(\text{shrK}\ B)\{\!|Kbc, C, Nb|\!\},\\ &\text{Crypt}(\text{shrK}\ B)\{\!|Kab, A, Nb|\!\},\ R|\!\}\end{aligned}$$

may add Says B A R

Inductively Modelling the Server, 1

1. If Kab is a fresh key (not used in evs) then

$$\begin{aligned}
&(\,\mathsf{Hash}_{\mathsf{shrK}\ A}\{\!|A, B, Na, -|\!\}, && \text{(request)}\\
&\ \ \mathsf{Crypt}(\mathsf{shrK}\ A)\{\!|Kab, B, Na|\!\}, && \text{(response)}\\
&\ \ Kab) \quad \in \quad \mathsf{respond}\ evs && \text{(last key)}
\end{aligned}$$

Only if the hash can be verified

Inductively Modelling the Server, 2

2. If $(Pa, Ra, Kab) \in$ respond evs and Kbc is fresh and $Pa =$ Hash$_{\text{shrK}\ A}\{\!|A, B, Na, P|\!\}$ then

$$
\begin{array}{ll}
(\,\text{Hash}_{\text{shrK}\ B}\{\!|B, C, Nb, Pa|\!\}, & \text{(request)} \\
\{\!|\text{Crypt}(\text{shrK}\ B)\{\!|Kbc, C, Nb|\!\}, & \text{(response)} \\
\quad \text{Crypt}(\text{shrK}\ B)\{\!|Kab, A, Nb|\!\}, & \\
\quad Ra|\!\}, & \\
Kbc) \quad \in \quad \text{respond}\ evs & \text{(last key)}
\end{array}
$$

An Easy Proof: Long-Term Keys Aren't Lost

By induction over $(P, R, K') \in \text{respond}\, evs$:

$$K \in \text{parts}\{R\} \implies K \text{ is fresh}$$

By induction over $evs \in \text{recur}\, lost$:

$$K \in \text{parts}\, H \iff K \in lost$$

(any long-term key K found in traffic was lost initially)

Typically need two nested inductions

Unicity of Nonces

At most one hash in the history H contains

- the key of an uncompromised agent ($A \notin lost$)
- any specified nonce value, Na

$$\exists B'\, P'.\, \forall B\, P.$$

$$\text{Hash}\{\!|\text{Key}(\text{shrK}\ A), A, B, Na, P|\!\} \in \text{parts}\ H$$

$$\rightarrow B = B' \wedge P = P'$$

Unicity of Session Keys

At most *two* certificates in the response (R) contain

- any particular session key, Kab . . .
- made for two uncompromised agents ($A, B \notin lost$)

$$\exists A'\,B'.\,\forall A\,B\,N.$$

$$\text{Crypt}(\text{Key}(\text{shrK}\ A))\{\!|Kab, B, N|\!\} \in \text{parts}\{R\}$$

$$\rightarrow (A' = A \land B' = B) \lor (A' = B \land B' = A)$$

Secrecy

Essential lemma, for any session key Kab:

$$K \in \mathsf{analz}(\,\{Kab\} \cup H\,) \iff K = Kab \lor K \in \mathsf{analz}\, H$$

Guarantee between uncompromised agents A and B:

$$\mathsf{Crypt}(\mathsf{shrK}\, A)\{\!|Kab, B, N|\!\} \in \mathsf{parts}\, H \Longrightarrow Kab \notin \mathsf{analz}\, H$$

Nonces not involved in proofs

Difficulties involving Certificates

- Danger of re-ordering
- Need for explicitness: name of other agent
- Special treatment of first & last agents
- Complexity of respond's definition

  Simpler version: arbitrary lists of certificates

Limitations of the Proofs

- Authentication of B to A not proved
- Authentication of A to B not provable!
- No dynamic loss of long-term keys
- Encryption assumed secure
- Type confusion not considered (not relevant?)

Statistics

- Two weeks human effort for proofs
- 30 lemmas and theorems
- 135 tactic commands
- Under five minutes CPU time
- Savings from protocol's symmetries

Conclusions

- Inductive definitions can model non-trivial processes
- Nested inductions cause no problems
- Multiple session keys are no obstacle
- Many types of protocols can be analyzed
- Must distinguish abstract level from implementation