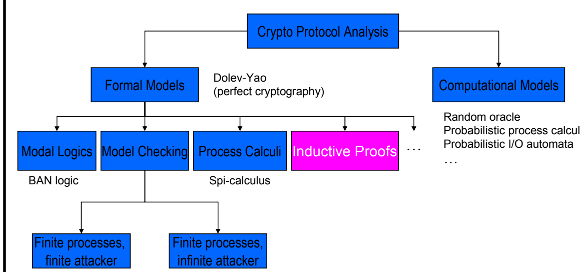
TECS Week 2005

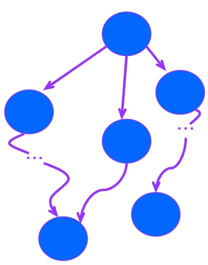
Protocol Verification by the Inductive Method

John Mitchell

Analysis Techniques



Recall: protocol state space



- Participant + attacker actions define a state transition graph
- A path in the graph is a trace of the protocol
- Graph can be
  - Finite if we limit number of agents, size of message, etc.
  - Infinite otherwise

[Paulson]

Analysis using theorem proving

- Correctness instead of bugs
  - Use higher-order logic to reason about possible protocol executions
- No finite bounds
  - Any number of interleaved runs
  - Algebraic theory of messages
  - No restrictions on attacker
- Mechanized proofs
  - Automated tools can fill in parts of proofs
  - Proof checking can prevent errors in reasoning

Inductive proofs

- Define set of traces
  - Given protocol, a trace is one possible sequence of events, including attacks
- Prove correctness by induction
  - For every state in every trace, no security condition fails
    - Works for safety properties only
  - Proof by induction on the length of trace

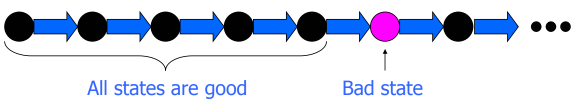
Two forms of induction

- Usual form for $\forall n \in Nat.\ P(n)$
  - Base case: $P(0)$
  - Induction step: $P(x) \Rightarrow P(x+1)$
  - Conclusion: $\forall n \in Nat.\ P(n)$
- Minimial counterexample form
  - Assume: $\exists x\ [\ \neg P(x) \wedge \forall y{<}x.\ P(y)\ ]$
  - Prove: contraction
  - Conclusion: $\forall n \in Nat.\ P(n)$

Both equivalent to "the natural numbers are well-ordered"

Use second form

- Given set of traces
  - Choose shortest sequence to bad state
  - Assume all steps before that OK
  - Derive contradiction
    - Consider all possible steps

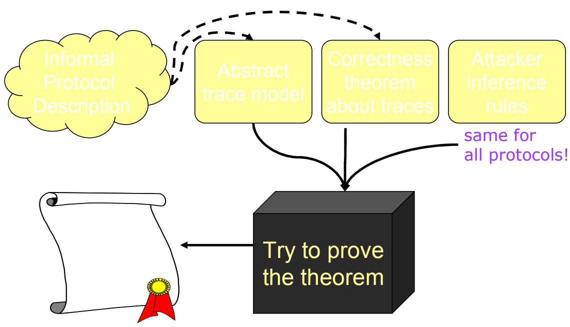


Sample Protocol Goals

- Authenticity: who sent it?
  - Fails if A receives message from B but thinks it is from C
- Integrity: has it been altered?
  - Fails if A receives message from B but message is not what B sent
- Secrecy: who can receive it?
  - Fails if attacker knows message that should be secret
- Anonymity
  - Fails if attacker or B knows action done by A

These are all safety properties

Inductive Method in a Nutshell



Work by Larry Paulson

- Isabelle theorem prover
  - General tool; protocol work since 1997
- Papers describing method
- Many case studies
  - Verification of SET protocol (6 papers)
  - Kerberos (3 papers)
  - TLS protocol
  - Yahalom protocol, smart cards, etc

http://www.cl.cam.ac.uk/users/lcp/papers/protocols.html

Verifying Security Protocols Using Isabelle - Microsoft Internet Explorer

http://www.cl.cam.ac.uk/users/lcp/papers/protocols.html

Verifying Security Protocols Using Isabelle

- Introductory papers
- Verification of the SET protocol
- Other results
- The original papers *(subsumed by the first paper below)*

Introductory papers

- L C Paulson. The inductive approach to verifying cryptographic protocols. *J. Computer Security* **6** (1998), 85–128.
- Giampaolo Bella
Inductive Verification of Cryptographic Protocols
PhD Thesis, University of Cambridge (2000)
- L C Paulson. Security protocols and their correctness.
Automated Reasoning Workshop 1998. St. Andrews, Scotland (1998)
Slides available
- L C Paulson. Proving security protocols correct.
IEEE Symposium on Logic in Computer Science. Trento, Italy (1999)
Slides available
- L C Paulson. Seven Years of Verifying Security Protocols
Schloß Dagstuhl Seminar 03451: *Applied Deductive Verification* (2003)
Slides available

Verification of the SET protocol

Isabelle

- Automated support for proof development
  - Higher-order logic
  - Serves as a logical framework
  - Supports ZF set theory & HOL
  - Generic treatment of inference rules
- Powerful simplifier & classical reasoner
- Strong support for inductive definitions

Agents and Messages

agent $A,B,\ldots$ = Server | Friend i | Spy
msg $X,Y,\ldots$ = Agent A
| Nonce N
| Key K
| $\{X, Y\}$
| Crypt $X\ K$

Typed, free term algebra, …

Protocol semantics

- 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

Define sets inductively

- Traces
  - Set of sequences of events
  - Inductive definition involves implications
    if $ev_1, \ldots, ev_n \in evs$, then add ev' to evs
- Information from a set of messages
  - parts H : parts of messages in H
  - analz H : information derivable from H
  - synth H : msgs constructible from H

Protocol events in trace

- Several forms of events
  - A sends B message X
  - A receives X
  - A stores X

$A \to B\ \{A, N_A\}_{pk(B)}$ — If ev is a trace and Na is unused, add `Says A B Crypt(pk B){A,Na}`

$B \to A\ \{N_B, N_A\}_{pk(A)}$ — If `Says A' B Crypt(pk B){A,X}` ∈ ev and Nb is unused, add `Says B A Crypt(pk A){Nb,X}`

$A \to B\ \{N_B\}_{pk(B)}$ — If `Says ...{X,Na}...` ∈ ev , add `Says A B Crypt(pk B){X}`

Dolev-Yao Attacker Model

- Attacker is a nondeterministic process
- Attacker can
  - Intercept any message, decompose into parts
  - Decrypt if it knows the correct key
  - Create new message from data it has observed
- Attacker cannot
  - Gain partial knowledge
  - Perform statistical tests
  - Stage timing attacks, …

Attacker Capabilities: Analysis

analz H is what attacker can learn from H

$X \in H \Rightarrow X \in \text{analz } H$

$\{X, Y\} \in \text{analz } H \Rightarrow X \in \text{analz } H$

$\{X, Y\} \in \text{analz } H \Rightarrow Y \in \text{analz } H$

Crypt $X\ K \in \text{analz } H$
& $K^{-1} \in \text{analz } H \Rightarrow X \in \text{analz } H$

Attacker Capabilities: Synthesis

synth H is what attacker can create from H

infinite set!

$X \in H \Rightarrow X \in \text{synth } H$

$X \in \text{synth } H$ & $Y \in \text{synth } H$

$\Rightarrow \{X, Y\} \in \text{synth } H$

$X \in \text{synth } H$ & $K \in \text{synth } H$

$\Rightarrow \text{Crypt } X\, K \in \text{synth } H$

Equations and implications

analz(analz H) = analz H
synth(synth H) = synth H
analz(synth H) = analz $H \cup$ synth H
synth(analz H) = ???

Nonce $N \in$ synth H $\Rightarrow$ Nonce $N \in H$
Crypt $K\, X \in$ synth H $\Rightarrow$ Crypt $K\, X \in H$
or $X \in$ synth H & $K \in H$

Attacker and correctness conditions

If $X \in$ synth(analz(spies *evs*)),
add *Says Spy B X*

X is not secret because attacker can construct it from the parts it learned from *events*

If *Says B A* $\{N_b, X\}_{pk(A)} \in evs$ &
Says A' B $\{N_b\}_{pk(B)} \in evs$,
Then *Says A B* $\{N_b\}_{pk(B)} \in evs$

If B thinks he's talking to A,
then A must think she's talking to B

Inductive Method: Pros & Cons

- Advantages
  - Reason about infinite runs, message spaces
  - Trace model close to protocol specification
  - Can "prove" protocol correct
- Disadvantages
  - Does not always give an answer
  - Failure does not always yield an attack
  - Still trace-based properties only
  - Labor intensive
    - Must be comfortable with higher-order logic

Caveat

- Quote from Paulson (J Computer Security, 2000)

The Inductive Approach to Verifying Cryptographic Protocols
  - The attack on the recursive protocol [40] is a sobering reminder of the limitations of formal methods… Making the model more detailed makes reasoning harder and, eventually, infeasible. A compositional approach seems necessary
- Reference
  - [40] P.Y.A. Ryan and S.A. Schneider, An attack on a recursive authentication protocol: A cautionary tale. *Information Processing Letters* 65, 1 (January 1998) pp 7 – 10.