Logics for Security Protocols

Anupam Datta
Fall 2007-08
Protocol Analysis Techniques

- Crypto Protocol Analysis
  - Formal Models
    - Dolev-Yao (perfect cryptography)
  - Computational Models
    - Random oracle
    - Probabilistic process calculi
    - Probabilistic I/O automata
- Model Checking
  - Murphi, AVISPA
- Protocol Logics
  - BAN, PCL
- Process Calculi
  - Applied II-calculus
- Inductive Proofs
  - Paulson, MSR
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
  • Proofs are very long
    - 4000 steps for Otway-Rees session key secrecy
Protocol Logics

◆ BAN Logic
   A Logic of Authentication by Michael Burrows, Martin Abadi, Roger Needham (1989)

◆ Historically, the first logic for reasoning about security protocols

◆ Syntax and proof system (axioms and rules) for proving authentication properties (semantics added in a later paper)
Advantages

- Proofs are relatively short (~ 2-3 pages)
  - cf. Paulson’s inductive proofs
- Proofs follow protocol design intuition
  - cf. model-checking, low-level theorem-proving
- Relatively easy to use
  - Still taught widely in security courses
- No explicit reasoning about traces and intruder
  - cf. Paulson’s inductive proofs
Disadvantages

- Not sound wrt now accepted model of protocol execution and attack
  - Protocols “proved” secure may be insecure
e.g. NS was proved secure using BAN
- Protocols are modeled using logical formulas (idealization step) as opposed to state machines or programs
- Many uses of non-standard logical concepts
  - Jurisdiction, control, “belief”, messages = propositions
- Only authentication properties, not secrecy
- Applicable to restricted classes of protocols

See Harper’s slides on BAN from 15-819 (linked from course web page)
Today

Protocol Composition Logic (PCL)
- Developed over the last few years (2001-07)
- Retain advantages of BAN; rectify deficiencies
- Semantic model similar to Paulson’s Inductive Method
- New proof techniques
  - Modular proofs
  - Cryptographic soundness

Reading tip
- Start from the example in Section 5 of the assigned reading
Protocol Composition Logic

- A logic for proving security of network protocols
- Illustrates use of programming language methods in computer security
  - Concurrency theory
    - Network protocols are concurrent programs
  - Floyd-Hoare style logic
    - Before-after assertions

15-812: Semantics of programming languages
Roadmap

◆ Intuition
◆ Formalism
  • Protocol programming language
  • Protocol logic
  • Proof System
◆ Example
  • Signature-based challenge-response
◆ Proof techniques

Formulated by Datta, Derek, Durgin, Mitchell, Pavlovic
Example: Challenge-Response

Alice reasons: if Bob is honest, then:
- only Bob can generate his signature
- if Bob generates a signature of the form \( \text{sig}_B\{m, n, A\} \),
  - he sends it as part of msg2 of the protocol, and
  - he must have received msg1 from Alice

Alice deduces: Received (B, msg1) ? Sent (B, msg2)
Formalizing the Approach

- Language for protocol description
  - Arrows-and-messages are informal.
- Protocol Operational Semantics
  - How does the protocol execute?
- Protocol logic
  - Stating security properties.
- Proof system
  - Formally proving security properties.
Protocol Programming Language

A protocol is described by specifying a "program" for each role
- Server = [receive x; new n; send {x, n}]

Building blocks
- Terms (think "messages")
  - names, nonces, keys, encryption, ...
- Actions (operations on terms)
  - send, receive, pattern match, ...
Terms

\[ t ::= \begin{align*}
  & c & \quad \text{constant term} \\
  & x & \quad \text{variable} \\
  & N & \quad \text{name} \\
  & K & \quad \text{key} \\
  & t, t & \quad \text{tupling} \\
  & \text{sig}_K\{t\} & \quad \text{signature} \\
  & \text{enc}_K\{t\} & \quad \text{encryption}
\end{align*} \]

Example: \( x, \text{sig}_B\{m, x, A\} \) is a term
**Actions**

- send $t$; send a term $t$
- receive $x$; receive a term into variable $x$
- match $t/p(x)$; match term $t$ against $p(x)$

- A program or cord is a sequence of actions

- Notation:
  - we often omit match actions
  - receive $\text{sig}_B\{A, n\} = \text{receive } x; \text{match } x/\text{sig}_B\{A, n\}$
Challenge-Response Programs

InitCR(A, X) = [
  new m;
  send A, X, {m, A};
  receive X, A, {x, sig_{X}{m, x, A}};
  send A, X, sig_{A}{m, x, X};
]

RespCR(B) = [
  receive Y, B, {y, Y};
  new n;
  send B, Y, {n, sig_{B}{y, n, Y}};
  receive Y, B, sig_{Y}{y, n, B};
]
Protocol Execution

- **Initial configuration**
  - Protocol is a finite set of roles
  - Set of principals and keys
  - Assignment of $\geq 1$ role to each principal

- **Run (trace)**

  A
  - new $x$
  - send $\{x\}_B$
  - receive $\{x\}_B$

  B
  - receive $\{x\}_B$
  - receive $\{z\}_B$
  - send $\{z\}_B$

  C
  - new $z$
  - send $\{z\}_B$

- Cord space is a multiset of cords
- Cords may react
  - via communication
  - via internal actions
- Sample reaction steps:
  - Communication:
    \[ [\text{send } t; S]_x \mid [\text{receive } x; T]_y \Rightarrow [S]_x \mid [T(t/x)]_y \]
  - Matching:
    \[ [\text{match } p(t)/p(x); S]_x \Rightarrow [S(t/x)]_x \]
Attacker capabilities

◆ Controls complete network
  • Can read, remove, inject messages
◆ Fixed set of operations on terms
  • Pairing
  • Projection
  • Encryption with known key
  • Decryption with known key
  • ...

Commonly referred to as “Dolev-Yao” attacker
PCL: Syntax

◆ Action formulas
  \[ a ::= \text{Send}(P,t) \mid \text{Receive}(P,t) \mid \text{Verify}(P,T) \mid \ldots \]

◆ Formulas
  \[ \varphi ::= a \mid \text{Has}(P,t) \mid \text{Honest}(N) \mid \neg \varphi \mid \varphi_1 \land \varphi_2 \mid \exists x \varphi \mid a < a \mid \ldots \]

◆ Modal formula
  \[ \varphi [ \text{actions} ]_P \varphi \]

◆ Example
  \[ \text{Has}(X, \text{secret}) \supset (X = A \lor X = B) \]

Specifying secrecy
Challenge-Response Property

**Specifying authentication for Initiator**

\[
\text{true \ } [\text{InitCR}(A, B) ]_A \text{ Honest}(B) \supset \\
( \\
\text{Send}(A, \{A,B,m\}) < \\
\text{Receive}(B, \{A,B,m\}) < \\
\text{Send}(B, \{B,A,{n, \text{sig}_B \{m, n, A}\}}) < \\
\text{Receive}(A, \{B,A,{n, \text{sig}_B \{m, n, A}\}}) \\
) \\
\]

**Semantics:** Property must hold in all protocol traces (similar to Paulson’s Inductive Method)
PCL: Semantics

- **Protocol Q**
  - Defines set of roles (e.g., initiator, responder)
  - Run R of Q is sequence of actions by principals following roles, plus attacker

- **Satisfaction**
  - \( Q, R \models \theta[\text{actions}]_P \phi \)
    - If some role of P in R does exactly actions starting from state where \( \theta \) is true, then \( \phi \) is true in state after actions completed irrespective of actions executed by other agents concurrently
  - \( Q \models \theta[\text{actions}]_P \phi \)
    - \( Q, R \models \theta[\text{actions}]_P \phi \) for all runs R of Q
Proof System

- **Goal:** formally prove security properties
- **Axioms**
  - Simple formulas provable by hand
- **Inference rules**
  - Proof steps
- **Theorem**
  - Formula obtained from axioms by application of inference rules
Sample axioms about actions

- **New data**
  - true [ new x ]_p Has(P,x)
  - true [ new x ]_p Has(Y,x) ⊃ Y=P

- **Actions**
  - true [ send m ]_p Send(P,m)

- **Verify**
  - true [ match x/sig_x{m} ]_p Verify(P,m)
Reasoning about knowledge

◆ Pairing
  • $\text{Has}(X, \{m,n\}) \supset \text{Has}(X, m) \land \text{Has}(X, n)$

◆ Encryption
  • $\text{Has}(X, \text{enc}_K(m)) \land \text{Has}(X, K^{-1}) \supset \text{Has}(X, m)$
Encryption and signature

- Public key encryption
  \( \text{Honest}(X) \land \text{Decrypt}(Y, \text{enc}_X\{m\}) \supset X=Y \)

- Signature
  \( \text{Honest}(X) \land \text{Verify}(Y, \text{sig}_X\{m\}) \supset \exists m' (\text{Send}(X, m') \land \text{Contains}(m', \text{sig}_X\{m\})) \)
Sample inference rules

◆ First-order logic rules
\[
\begin{array}{c}
\varphi \\
\hline 
\varphi \supset \theta \\
\hline
\theta
\end{array}
\]

◆ Generic rules
\[
\begin{array}{c}
\theta \left[ \text{actions} \right]_p \psi \\
\hline
\theta \left[ \text{actions} \right]_p \varphi
\end{array}
\]
\[
\begin{array}{c}
\theta \left[ \text{actions} \right]_p \psi \land \varphi
\end{array}
\]
Honesty rule

∀ roles R of Q. ∀ protocol steps A of R.

\[
\begin{align*}
\text{Start}(X) \ [ \ ]_X \phi & \quad \phi \ [ \ A \ ]_X \phi \\
\hline
Q \models \text{Honest}(X) \supset \phi
\end{align*}
\]

• Example use:
  - If Y receives a message m from X, and
  - Honest(X) \supset (Sent(X,m) \supset Received(X,m'))
  - then Y can conclude
    Honest(X) \supset Received(X,m'))
Correctness of CR

\[
\text{InitCR}(A, X) = \left[ \begin{array}{l}
\text{new } m; \\
\text{send } A, X, \{m, A\}; \\
\text{receive } X, A, \{x, \text{sig}_X\{m, x, A\}\}; \\
\text{send } A, X, \text{sig}_A\{m, x, X\}; \\
\end{array} \right]
\]

\[
\text{RespCR}(B) = \left[ \begin{array}{l}
\text{receive } Y, B, \{y, Y\}; \\
\text{new } n; \\
\text{send } B, Y, \{n, \text{sig}_B\{y, n, Y\}\}; \\
\text{receive } Y, B, \text{sig}_Y\{y, n, B\}; \\
\end{array} \right]
\]

\[
\text{CR |- true} \ [\text{InitCR}(A, B)]_A \text{ Honest}(B) \Rightarrow
\begin{align*}
\text{Send}(A, \{A,B,m\}) < \\
\text{Receive}(B, \{A,B,m\}) < \\
\text{Send}(B, \{B,A,\{n, \text{sig}_B\{m, n, A\}\}\}) < \\
\text{Receive}(A, \{B,A,\{n, \text{sig}_B\{m, n, A\}\}\})
\end{align*}
\]
Correctness of CR – step 1

\[\text{InitCR}(A, X) = [\]
\[
\begin{align*}
&\quad \text{new } m; \\
&\quad \text{send } A, X, \{m, A\}; \\
&\quad \text{receive } X, A, \{x, \text{sig}_X\{m, x, A\}\}; \\
&\quad \text{send } A, X, \text{sig}_A\{m, x, X\}; \\
\end{align*}
\]

\[\text{RespCR}(B) = [\]
\[
\begin{align*}
&\quad \text{receive } Y, B, \{y, Y\}; \\
&\quad \text{new } n; \\
&\quad \text{send } B, Y, \{n, \text{sig}_B\{y, n, Y\}\}; \\
&\quad \text{receive } Y, B, \text{sig}_y\{y, n, B\}; \\
\end{align*}
\]

1. A reasons about her own actions

\[\text{CR} \vdash \text{true [ InitCR}(A, B) ]_A \]

\[\text{Verify}(A, \text{sig}_B \{m, n, A\})\]
Correctness of CR - step 2

\[ \text{InitCR}(A, X) = [ \]
\[ \text{new } m; \]
\[ \text{send } A, X, \{m, A\}; \]
\[ \text{receive } X, A, \{x, \text{sig}_X\{m, x, A\}\}; \]
\[ \text{send } A, X, \text{sig}_A\{m, x, X\}; \]
\[ ] \]

\[ \text{RespCR}(B) = [ \]
\[ \text{receive } Y, B, \{y, Y\}; \]
\[ \text{new } n; \]
\[ \text{send } B, Y, \{n, \text{sig}_B\{y, n, Y\}\}; \]
\[ \text{receive } Y, B, \text{sig}_Y\{y, n, B\}; \]
\[ ] \]

2. Properties of signatures

\[ \text{CR} |- \text{true } [ \text{InitCR}(A, B) ]_A \text{ Honest}(B) \supset \]
\[ \exists m' (\text{Send}(B, m') \land \text{Contains}(m', \text{sig}_B \{m, n, A\}) \]

Recall signature axiom
Correctness of CR - Honesty

\[
\begin{align*}
\text{InitCR}(A, X) &= [ \\
&\quad \text{new } m; \\
&\quad \text{send } A, X, \{m, A\}; \\
&\quad \text{receive } X, A, \{x, \text{sig}_X \{m, x, A\}\}; \\
&\quad \text{send } A, X, \text{sig}_A \{m, x, X\}; \\
&\quad ] \nonumber
\end{align*}
\]

\[
\begin{align*}
\text{RespCR}(B) &= [ \\
&\quad \text{receive } Y, B, \{y, Y\}; \\
&\quad \text{new } n; \\
&\quad \text{send } B, Y, \{n, \text{sig}_B \{y, n, Y\}\}; \\
&\quad \text{receive } Y, B, \text{sig}_y \{y, n, B\}; \\
&\quad ] \nonumber
\end{align*}
\]

Invariant proved with Honesty rule

\[
\begin{align*}
\text{CR} \vdash \& \text{Honest}(X) \land \\
&\text{Send}(X, m') \land \text{Contains}(m', \text{sig}_X \{y, x, Y\}) \land \neg \text{New}(X, y) \Rightarrow \\
m = X, Y, \{x, \text{sig}_B \{y, x, Y\}\} \land \text{Receive}(X, \{Y, X, \{y, Y\}\})
\end{align*}
\]

Induction over protocol steps
Correctness of CR - step 3

\[
\text{InitCR}(A, X) = [ \\
\quad \text{new } m; \\
\quad \text{send } A, X, \{m, A\}; \\
\quad \text{receive } X, A, \{x, \text{sig}_X\{m, x, A\}\}; \\
\quad \text{send } A, X, \text{sig}_A\{m, x, X\}; \\
\]
\]

\[
\text{RespCR}(B) = [ \\
\quad \text{receive } Y, B, \{y, Y\}; \\
\quad \text{new } n; \\
\quad \text{send } B, Y, \{n, \text{sig}_B\{y, n, Y\}\}; \\
\quad \text{receive } Y, B, \text{sig}_Y\{y, n, B\}; \\
\]
\]

3. Use Honesty invariant

\[
\text{CR} \vdash \text{true } [\text{InitCR}(A, B) ]_A \text{ Honest}(B) \supset \text{Receive}(B, \{A, B, m\}), ...
\]
Correctness of CR - step 4

InitCR(A, X) = [
    new m;
    send A, X, {m, A};
    receive X, A, {x, sig_X{m, x, A}};
    send A, X, sig_A{m, x, X};
]

RespCR(B) = [
    receive Y, B, {y, Y};
    new n;
    send B, Y, {n, sig_B{y, n, Y}};
    receive Y, B, sig_Y{y, n, B};
]

4. Use properties of nonces for temporal ordering

\[ CR \vdash true [ \text{InitCR}(A, B) ]_A \quad \text{Honest}(B) \Rightarrow \text{Auth} \]

Nonces are “fresh” random numbers
We have a proof. So what?

◆ Soundness Theorem:
  • if Q |- φ then Q |= φ
  • If φ is a theorem then φ is a valid formula
◆ φ holds in any step in any run of protocol Q
  • Unbounded number of participants
  • Dolev-Yao intruder
Thanks!

Questions?