Protocol Composition Logic

John Mitchell
Stanford

Intuition

Reason about local information
- I chose a new number
- I sent it out encrypted
- I received it decrypted
- Therefore: someone decrypted it

Incorporate knowledge about protocol
- Protocol: Server only answers if sent a request
  - If server not corrupt and
    - I receive an answer from the server, then
    - the server must have received a request

Intuition: Picture

Alice's information
- Protocol
- Private data
- Sends and receives

Example: Challenge-Response

Alice reasons: if Bob is honest, then:
- only Bob can generate his signature. [protocol independent]
- 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. [protocol dependent]
- Alice deduces: Received (B, msg1) \& Sent (B, msg2)

Formalizing the Approach

Language for protocol description
- Write program for each role of protocol

Protocol logic
- State security properties
- Specialized form of temporal logic

Proof system
- Formally prove security properties
- Supports modular proofs

Cords

Protocol programming language
- Server = \{receive x; new n; send \{x, n\}\}

Building blocks
- Terms
  - names, nonces, keys, encryption, ...
- Actions
  - send, receive, pattern match, ...
**Terms**

- \( t \) ::= \( c \) constant term
- \( x \) variable
- \( N \) name
- \( K \) key
- \( t, t \) tupling
- \( \text{sig}_K(t) \) signature
- \( \text{enc}_K(t) \) encryption

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

**Actions and Cords**

- **Actions**
  - send \( t \): send a term \( t \)
  - receive \( x \): receive a term into variable \( x \)
  - match \( t/p(x) \): match term \( t \) against \( p(x) \)
- **Cord**
  - Sequence of actions
- **Notation**
  - Some match actions are omitted in slides
  - \( \text{receive sig}_B(A, n) \) means \( \text{receive } x; \text{match } x/\text{sig}_B(A, n) \)

**Challenge-Response as Cords**

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

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

**Execution Model**

- **Protocol**
  - Cord gives program for each protocol role
- **Initial configuration**
  - Set of principals and keys
  - Assignment of \( \geq 1 \) role to each principal
- **Run**

```
A
  ∨
new x send \{x\}B
  receive \{x\}B
B
  ∨
new n send \{n\}B
  receive \{n\}B
```

**Formulas true at a position in run**

- **Action formulas**
  - \( a ::= \text{Send}(P, m) \mid \text{Receive}(P, m) \mid \text{New}(P, t) \mid \text{Decrypt}(P, t) \mid \text{Verify}(P, t) \)
- **Formulas**
  - \( \varphi ::= a \mid \text{Has}(P, t) \mid \text{Fresh}(P, t) \mid \text{Honest}(N) \mid \text{Contains}(t_1, t_2) \mid \neg \varphi \mid \varphi_1 \land \varphi_2 \mid \exists x \varphi \mid \Box \varphi \mid \Diamond \varphi \)
- **Example**
  - \( \text{After}(a, b) = \Diamond (b \land \Box \Diamond a) \)

**Modal Formulas**

- **After actions, postcondition**
  - \( [\text{actions}]_P \varphi \) where \( P = \langle \text{princ}, \text{role id} \rangle \)
- **Before/after assertions**
  - \( \varphi [\text{actions}]_P \psi \)
- **Composition rule**
  - \( \frac{\varphi [S]_P \psi \quad \psi [T]_P \theta}{\varphi [ST]_P \theta} \) Note: same \( P \) in all formulas
Security Properties

Authentication for Initiator

\[ \text{GR} \models [\text{InitGR}(A, B)], \text{Honest}(B) \supset \]
\[ \text{ActionsInOrder}(\]
\[ \text{Send}(A, (A, B, m)), \]
\[ \text{Receive}(B, (A, B, m)), \]
\[ \text{Send}(B, (B, A, (n, \text{sig}_B(m, n, A)))), \]
\[ \text{Receive}(A, (B, A, (n, \text{sig}_B(m, n, A)))) \]
\]

Shared secret

\[ \text{NS} \models [\text{InitNS}(A, B), \text{Honest}(B) \supset (\text{Has}(X, m) \supset X=A \land X=B) \]

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

\[ Q, R \models [\text{actions}], \phi \]
\[ \text{Some role of } P \text{ in } R \text{ does exactly } \text{actions and } \phi \text{ is true in state after } \text{actions completed} \]

\[ Q, R \models [\text{actions}], \phi \text{ for all runs } R \text{ of } Q \]

Proof System

Goal: prove properties formally

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

- [new x] P Has(P, x)
- [new x] P Has(Y, x) \supset Y=P

Actions

- [send m] P \diamond Send(P, m)

Knowledge

- [receive m] P Has(P, m)

Verify

- [match x/sig\_x(m)] P \diamond Verify(P, m)

Reasoning about knowledge

Pairing

- Has(X, (m, n)) \supset Has(X, m) \land Has(X, n)

Encryption

- Has(X, enc\_x(m)) \land Has(X, K^{-1}) \supset Has(X, m)

Encryption and signature

Public key encryption

- Honest(X) \land \diamond Decrypt(Y, enc\_x(m)) \supset X=Y

Signature

- Honest(X) \land \diamond Verify(Y, sig\_x(m)) \supset \exists m' (\diamond Send(X, m') \land Contains(m', sig\_x(m)))
Sample inference rules

\[ \text{Preservation rules} \]
\[ \psi \left[ \text{actions} \right] \models \text{Has}(X, t) \]
\[ \psi \left[ \text{actions}; \text{action} \right] \models \text{Has}(X, t) \]

\[ \text{Generic rules} \]
\[ \psi \left[ \text{actions} \right] \models \phi \]
\[ \psi \left[ \text{actions} \right] \models \phi \land \psi \]

Bidding conventions (motivation)

\[ \text{Blackwood response to 4NT} \]
\[ -5\spadesuit : 0 \text{ or } 4 \text{ aces} \]
\[ -5\clubsuit : 1 \text{ ace} \]
\[ -5\heartsuit : 2 \text{ aces} \]
\[ -5\diamondsuit : 3 \text{ aces} \]

\[ \text{Reasoning} \]
\[ - \text{If my partner is following Blackwood,} \]
\[ \text{then if she bid } 5\heartsuit, \text{ she must have } 2 \text{ aces} \]

Honesty rule (rule scheme)

\[ \forall \text{roles } R \text{ of } Q. \forall \text{ initial segments } A \subseteq R. \]
\[ Q \models [A, \phi] \]
\[ Q \models \text{Honest}(X) \models \phi \]

- This is a finitary rule:
  - Typical protocol has 2-3 roles
  - Typical role has 1-3 receives
  - Only need to consider A waiting to receive

Honesty rule (example use)

\[ \forall \text{roles } R \text{ of } Q. \forall \text{ initial segments } A \subseteq R. \]
\[ Q \models [A, \phi] \]
\[ Q \models \text{Honest}(X) \models \phi \]

- Example use:
  - If Y receives a message from X, and
  \[ \text{Honest}(X) \models (\text{Sent}(X, m) \models \text{Received}(X, m')) \]
  then Y can conclude
  \[ \text{Honest}(X) \models \text{Received}(X, m')) \]

Correctness of CR

\[ \text{InitCR}(A, X) \left\{ \begin{array}{l}
\text{new } m; \\
\text{send } A, X, \{m, A\}; \\
\text{receive } X, A, \{x, \text{sig}(m, x, A)\}; \\
\text{send } A, X, \text{sig}(m, x, X); \\
\text{receive } Y, B, \{y, \text{sig}(y, n, B)\};
\end{array} \right\} \]
\[ \text{RespCR}(B) \left\{ \begin{array}{l}
\text{new } n; \\
\text{receive } Y, B, \{y, Y\}; \\
\text{new } n; \\
\text{send } B, Y, \{n, \text{sig}(y, n, Y)\}; \\
\text{send } A, X, \text{sig}(m, x, X); \\
\text{receive } Y, B, \text{sig}(y, n, B); \\
\end{array} \right\} \]

\[ \text{CR} \models [\text{InitCR}(A, B)] \}_{A} \]
\[ \text{Honest(B)} \models \text{ActionsInOrder} \]
\[ \text{Send}(A, \{A,B,m\}); \\
\text{Receive}(B, \{A,B,m\}); \\
\text{Send}(B, \{B,A,m, \text{sig}(m, n, A)\}); \\
\text{Receive}(A, \{B,A,m, \text{sig}(m, n, A)\}); \]

Correctness of CR - step 1

\[ \text{InitCR}(A, X) \left\{ \begin{array}{l}
\text{new } m; \\
\text{send } A, X, \{m, A\}; \\
\text{receive } X, A, \{x, \text{sig}(m, x, A)\}; \\
\text{send } A, X, \text{sig}(m, x, X); \\
\text{receive } Y, B, \{y, \text{sig}(y, n, B)\};
\end{array} \right\} \]
\[ \text{RespCR}(B) \left\{ \begin{array}{l}
\text{new } n; \\
\text{receive } Y, B, \{y, Y\}; \\
\text{new } n; \\
\text{send } B, Y, \{n, \text{sig}(y, n, Y)\}; \\
\text{send } A, X, \text{sig}(m, x, X); \\
\text{receive } Y, B, \text{sig}(y, n, B); \\
\end{array} \right\} \]

1. A reasons about it's own actions
\[ \text{CR} \models [\text{InitCR}(A, B)] \}_{A} \]
\[ \diamond \text{Verify}(A, \text{sig}(m, n, A)) \]
Correctness of CR – step 2

2. Properties of signatures

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

Correctness of CR – step 3

3. Use Honesty rule

\[ CR \vdash [\text{InitCR}(A, B)] \land \text{Honest}(B) \supset \Diamond \text{Receive}(B, (A, B, m)) \]

Correctness of CR – step 4

4. Use properties of nonces for temporal ordering

\[ CR \vdash [\text{InitCR}(A, B)] \land \text{Honest}(B) \supset \text{Auth} \]

Complete proof

What does proof tell us?

◆ Soundness Theorem:
  • If \( Q \vdash \phi \) then \( Q \vdash \phi \)
  • If \( \phi \) is provable about protocol \( Q \), then \( \phi \) is true about protocol \( Q \).
◆ \( \phi \) true in every run of \( Q \)
  • Dolev-Yao intruder
  • Unbounded number of participants
Weak Challenge-Response

InitWCR(A, X) = 
new m;
send A, X, {m};
receive X, A, {x, sigx{m, x}};
send A, X, sigA{m, x}};

RespWCR(B) = 
receive Y, B, {y};
new n;
send B, Y, {n, sigB{y, n}};
receive Y, B, sigY{y, n}};

Correctness of WCR – step 1

1. A reasons about it’s own actions

WCR |- [ InitWCR(A, B ) ]^A
\diamond Verify(A, sigB{m, n})

Correctness of WCR – step 2

2. Properties of signatures

CR |- [ InitCR(A, B ) ]^A Honest(B) \Rightarrow
\exists m’ (\diamond Send(B, m’) \land Contains(m’, sigB{m, n, A}))

Correctness of WCR – Honesty

Honesty invariant

CR |- Honest(X) \land
\diamond Send(X, m’ ) \land Contains(m’, sigy{y, x}) \Rightarrow
\diamond New(X, y ) \Rightarrow
m’ \land X, (x, sigB{y, x}) \land \diamond Receive(X, (Z, X, (y, Z)))

Correctness of WCR – step 3

3. Use Honesty rule

WCR |- [ InitWCR(A, B ) ]^A Honest(B) \Rightarrow
\diamond Receive(B, (Z,B,m))

Result

\diamond WCR does not have the strong authentication property for the initiator

\diamond Counterexample

- Intruder can forge senders and receivers identity in first two messages
  - A \rightarrow X(B) m
  - X(C) \rightarrow B m
  - B \rightarrow X(C) n, sigB{m, n}
  - X(B) \rightarrow A n, sigB{m, n}
Extensions

◆ Add Diffie-Hellman primitive
  • Can prove authentication and secrecy for key exchange protocols (STS, ISO-97898-3)

◆ Add symmetric encryption, hashing
  • Can prove authentication for ISO-9798-2, SKID3

Composition Rules

◆ Prove assertions from invariants
  \[ \Gamma |- \phi [...] \psi \]

◆ Invariant weakening rule
  \[ \Gamma |- \phi [...] \psi \]
  \[ \Gamma \cup \Gamma' |- \phi [...] \psi \]
  If combining protocols, extend assertions to combined invariants

◆ Prove invariants from protocol
  \[ Q \vdash \Gamma \vdash Q' \vdash \Gamma \]
  Use honesty (invariant) rule to show that both protocols preserve assumed invariants

Combining protocols

\[ \Gamma \vdash \text{Honest}(X) \Rightarrow \ldots CR \vdash \text{Honest}(X) \Rightarrow \ldots \]
\[ \Gamma |- \text{Secrecy} \quad \Gamma' |- \text{Authentication} \]
\[ \Gamma \cup \Gamma' |- \text{Secrecy} \quad \Gamma \cup \Gamma' |- \text{Authentication} \]
\[ \Gamma \cup \Gamma' |- \text{Secrecy} \wedge \text{Authentication} \]
\[ \text{DH} \circ \text{CR} \vdash \Gamma \cup \Gamma' \]
\[ \text{ISO} \vdash \text{Secrecy} \wedge \text{Authentication} \]

Protocol Templates

◆ Protocols with function variables instead of specific operations
  • One template can be instantiated to many protocols

◆ Advantages:
  • proof reuse
  • design principles/patterns

Extending Formalism

◆ Language Extension
  • Add function variables to term language for cords and logic (HOL)

◆ Semantics
  • \[ Q |- \phi \Leftrightarrow \sigma Q |- \sigma \phi, \text{for all substitutions } \sigma \]
  eliminating all function variables

◆ Soundness Theorem
  • Every provable formula is valid

Example

Challenge-Response Template

\[
\begin{align*}
A &\rightarrow B: m \\
B &\rightarrow A: n, \Gamma(B,A,n,m) \\
A &\rightarrow B: c(A,B,n,m)
\end{align*}
\]

ISO-9798-2   SKID3   ISO-9798-3

Abstraction

Instantiation
Proof Structure

Template

- axiom
- hypothesis

Instance

- Discharge hypothesis

Modular proof techniques (2)

- Combining protocol templates
  - If protocol P is a hypotheses-respecting instance of two different templates, then it has the properties of both.
- Benefits:
  - Modular proofs of properties
  - Formalization of protocol refinements

Refinement Example Revisited

Encrypt Signatures

| A → B: g^a, A |
| B → A: g^b, E_x (sig_A (g^a, g^b, A)) |
| A → B: E_x (sig_A (g^a, g^b, B)) |

- Two templates:
  - Template 1: authentication + shared secret
    - (Preserves existing properties; proof reused)
  - Template 2: identity protection (encryption)
    - (Adds new property)

Authenticated key exchange

- AKE1
  - A → B: g^a, A
  - B → A: g^b, (f(B, A, g^b, g^a))
  - A → B: G(A, B, g^a, g^b)

- AKE2
  - A → B: g^a
  - B → A: g^b, (f(B, g^b, g^a), f'(B, g^b, g^a))
  - A → B: G(A, g^a, g^b), G'(A, g^a, g^b)

Sample projects using this method

- Key exchange
  - STS family, JFK, IKEv2
  - Diffie-Hellman -> MQV
  - GDOI [Meadows, Pavlovic]
- Work in progress, mostly done
  - SSL verification
  - Wireless 802.11i
- Implementation of logic
  - Student project, using Isabelle

Symbolic vs Computational model

- Suppose \( \Gamma \vdash [\text{actions}]_X \phi \)
  - If a protocol P satisfies invariants \( \Gamma \), then if X does actions, \( \phi \) will be true
- Symbolic soundness
  - No idealized adversary acting against "perfect" cryptography can make \( \phi \) fail
- Computational soundness
  - No probabilistic polytime adversary can make \( \phi \) fail with nonnegligible probability

H. Krawczyk: The Cryptography of the IPSec and IKE Protocols [CRYPTO'03]