Contributions

- **Protocol derivation**
  - Build security protocols by combining parts from standard sub-protocols.

- **Proof of correctness**
  - Prove protocols correct using logic that follows steps of derivation.
Outline

- Derivation System [CSFW03]
  - Motivating examples
  - Main concepts
  - Benefits
- Compositional Logic [CSFW01, CSFW03]
- Formalizing Composition [MFPS03]
- Formalizing Refinements [CSFW04]
- Conclusions and Future Work
Example

Construct protocol with properties:
- Shared secret
- Authenticated
- Identity Protection
- DoS Protection

Design requirements for **IKE, JFK, IKEv2** (IPSec key exchange protocol)
Component 1

**Diffie Hellman**

\[
\begin{align*}
A & \rightarrow B: g^a \\
B & \rightarrow A: g^b
\end{align*}
\]

- **Shared secret (with someone)**
  - A deduces:
    
    \[
    \text{Knows}(Y, g^{ab}) \supset (Y = A) \lor \text{Knows}(Y, b)
    \]

- Authenticated
- Identity Protection
- DoS Protection
Component 2

Challenge-Response

A $\rightarrow$ B: m, A
B $\rightarrow$ A: n, $\text{sig}_B \{m, n, A\}$
A $\rightarrow$ B: $\text{sig}_A \{m, n, B\}$

- Shared secret
- Authenticated
  - A deduces: Received (B, msg1) $\wedge$ Sent (B, msg2)
- Identity Protection
- DoS Protection
Composition

ISO-9798-3

- Shared secret: $g^{ab}$
- Authenticated
- Identity Protection
- DoS Protection

$m := g^a$

$n := g^b$

\[
\begin{align*}
A \rightarrow B & : g^a, A \\
B \rightarrow A & : g^b, \text{sig}_B \{g^a, g^b, A\} \\
A \rightarrow B & : \text{sig}_A \{g^a, g^b, B\}
\end{align*}
\]
Refinement

Encrypt Signatures

\begin{align*}
A \rightarrow B &: \ g^a, A \\
B \rightarrow A &: \ g^b, E_K \{\text{sig}_B \{g^a, g^b, A\}\} \\
A \rightarrow B &: \ E_K \{\text{sig}_A \{g^a, g^b, B\}\}
\end{align*}

- Shared secret: $g^{ab}$
- Authenticated
- Identity Protection
- DoS Protection
Transformation

Use cookie: JFK core protocol

A → B:  \( g^a, A \)
B → A:  \( g^b, \text{hash}_{KB}\{g^b, g^a\} \)
A → B:  \( g^a, g^b, E_K\{\text{sig}_A\{g^a, g^b, B\}\}, \text{hash}_{KB}\{g^b, g^a\} \)
B → A:  \( g^b, E_K\{\text{sig}_B\{g^a, g^b, A\}\} \)

- Shared secret: \( g^{ab} \)
- Authenticated
- Identity Protection
- DoS Protection
Derivation Framework

- Protocols are constructed from:
  - components

by applying a series of:
- composition, refinement and transformation operations.

- Properties accumulate as a derivation proceeds.

- Examples:
  - STS, ISO-9798-3, JFKi, JFKr, IKE, GDOI, Kerberos, Needham-Schroeder,...
Benefits and Directions

- Modular analysis of protocols.
- Organization of protocols into taxonomies.
- Underpin protocol design principles and patterns.
- Protocol synthesis.
Outline

- Derivation System
- **Compositional Logic** [CSFW01,CSFW03]
  - Main idea
  - Syntax, semantics and proof system
- Formalizing Composition
- Formalizing Refinements
- Conclusions and Future Work
Protocol Logic: Main idea

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

Honest Principals, Attacker

 Protocol

Private Data
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 msg 2 of the protocol and
  - he must have received msg1 from Alice. [protocol specific]

Alice deduces: Received $(B, \text{msg1}) \land \text{Sent } (B, \text{msg2})$
Protocol
- "Program" for each protocol role

Initial configuration
- Set of principals and key
- Assignment of $\geq 1$ role to each principal

Run

```
Position in run
```

```
A  new x  send{x}_B  
    |      |      |
    v      v      v
B  recv{x}_B  recv{z}_B  
    |      |      |
    v      v      v
C  new z  send{z}_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**
  \[
  \varphi ::= a \mid \text{Has}(P,t) \mid \text{Fresh}(P,t) \mid \text{Honest}(N) \\
  \mid \text{Contains}(t_1, t_2) \mid \neg \varphi \mid \varphi_1 \land \varphi_2 \mid \exists x \ \varphi \\
  \mid o \varphi \mid \Diamond \varphi
  \]

- **Example**
  \[
  \text{After}(a,b) = \Diamond (b \land o \Diamond a)
  \]
Modal Formulas

- **After actions, postcondition**
  
  \[ [ \text{actions} ]_P \varphi \quad \text{where} \quad P = \langle \text{princ, role id} \rangle \]
  
  - If \( P \) does ‘actions’, starting from initial state, then \( \varphi \) holds in resulting state

- **Before/after assertions**
  
  \( \varphi \ [ \text{actions} ]_P \psi \)
  
  - If \( \varphi \) holds in some state, and \( P \) does ‘actions’, then \( \psi \) holds in resulting state
Diffie-Hellman: Property

- **Formula**
  - $[\text{new } a]_A \text{Fresh}(A, g^a)$

- **Explanation**
  - Modal form: $[\text{actions}]_p \varphi$
  - Actions: $[\text{new } a]_A$
  - Postcondition: Fresh$(A, g^a)$
Challenge Response: Property

- Modal form: $\varphi [ \text{actions} ]_P \psi$
  - precondition: $\text{Fresh}(A, m)$
  - actions: $[ \text{Initiator role actions} ]_A$
  - postcondition:
    $\text{Honest}(B) \Rightarrow \text{ActionsInOrder}(
    \text{send}(A, \{A, B, m\}),$
    \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\}\}\})
  )
Proof System

- **Sample Axioms:**
  - **Reasoning about knowledge:**
    - $[\text{receive } m ]_A \text{ Has}(A,m)$
    - $\text{Has}(A, \{m,n\}) \supset \text{Has}(A, m) \land \text{Has}(A, n)$
  - **Reasoning about crypto primitives:**
    - $\text{Honest}(X) \land \Diamond \text{Decrypt}(Y, \text{enc}_X\{m\}) \supset X=Y$
    - $\text{Honest}(X) \land \Diamond \text{Verify}(Y, \text{sig}_X\{m\}) \supset \exists m' (\Diamond \text{Send}(X, m') \land \text{Contains}(m', \text{sig}_X\{m\}))$

- **Soundness Theorem:**
  Every provable formula is valid
Outline

- Derivation System
- Compositional Logic
- Formalizing Composition [MFPS03]
- Formalizing Refinements
- Conclusions and Future Work
Central Issues

- **Additive Combination:**
  - Accumulate security properties of combined parts, assuming they do not interfere
    - In logic: *before-after assertions*

- **Non-destructive Combination:**
  - Ensure combined parts do not interfere
    - In logic: *invariance assertions*
Proof steps (Intuition)

- Protocol independent reasoning
  - $\text{Has}(A, \{m,n\}) \supset \text{Has}(A, m) \land \text{Has}(A, n)$
  - Still good: unaffected by composition

- Protocol specific reasoning
  - “if honest Bob generates a signature of the form $\text{sig}_B \{m, n, A\}$,
    - he sends it as part of msg 2 of the protocol and
    - he must have received msg1 from Alice”
  - Could break: Bob’s signature from one protocol could be used to attack another

Protocol-specific proof steps use invariants
Invariants

- Reasoning about honest principals
  - Invariance rule, called “honesty rule”
- Preservation of invariants under composition
  - If we prove $\text{Honest}(X) \supset \varphi$ for protocol 1 and compose with protocol 2, is formula still true?
Honesty Rule

Definition

- A basic sequence of actions begins with receive, ends before next receive

Rule

\[ [\ ]_X \varphi \quad \text{For all } B \in \text{BasicSeq}(Q). \varphi [B]_X \varphi \]

\[
Q \Rightarrow \text{Honest}(X) \supset \varphi
\]

Example

CR \Rightarrow \text{Honest}(X) \supset

\( (\text{Sent}(X, m_2) \supset \text{Recd}(X, m_1)) \)
Composing protocols

\[ \Gamma \vdash \text{Secrecy} \]

\[ \Gamma \cup \Gamma' \vdash \text{Secrecy} \]

\[ \Gamma \cup \Gamma' \vdash \text{Secrecy} \land \text{Authentication} \ [\text{additive}] \]

\[ \text{DH} \triangleright \text{Honest}(X) \supset \ldots \]

\[ \Gamma' \vdash \text{Authentication} \]

\[ \Gamma \cup \Gamma' \vdash \text{Authentication} \]

\[ \text{DH} \bullet \text{CR} \triangleright \Gamma \cup \Gamma' \ [\text{nondestructive}] \]

\[ \text{ISO} \triangleright \text{Secrecy} \land \text{Authentication} \]
Composition Rules

- Invariant weakening rule
  \[ \Gamma \vdash \phi \quad \psi \]
  \[ \Gamma \cup \Gamma' \vdash \phi \quad \psi \]

- Sequential Composition
  \[ \Gamma \vdash \phi \quad [S] \psi \quad \Gamma \vdash [T] \psi \]
  \[ \Gamma \vdash \phi \quad [ST] \psi \]

- Prove invariants from protocol
  \[ Q \uparrow \Gamma \quad Q' \uparrow \Gamma \]
  \[ Q \cdot Q' \uparrow \Gamma \]
Outline

- Derivation System
- Compositional Logic
- Formalizing Composition
- Formalizing Refinements [CSFW04]
- Conclusions and Future Work
Protocol Templates

- Protocols with function variables instead of specific cryptographic operations (Higher-order extension of protocol logic)
- **Idea:** One template can be instantiated to many protocols
- **Advantages:**
  - proof reuse
  - design principles/patterns
Example

Challenge-Response Template

A $\rightarrow$ B:  $m$
B $\rightarrow$ A:  n, $F(B,A,n,m)$
A $\rightarrow$ B:  $G(A,B,n,m)$

A $\rightarrow$ B:  $m$
B $\rightarrow$ A:  n, $E_{KAB}(n,m,B)$
A $\rightarrow$ B:  $E_{KAB}(n,m)$

A $\rightarrow$ B:  $m$
B $\rightarrow$ A:  n, $H_{KAB}(n,m,B)$
A $\rightarrow$ B:  $H_{KAB}(n,m,A)$

A $\rightarrow$ B:  $m$
B $\rightarrow$ A:  n, $\text{sig}_B(n,m,A)$
A $\rightarrow$ B:  $\text{sig}_A(n,m,B)$

ISO-9798-2  SKID3  ISO-9798-3
Abstraction-Instantiation Method (1)

- Characterizing protocol concepts
  - Step 1: Under hypotheses about function variables and invariants, prove security property of template
  - Step 2: Instantiate function variables to cryptographic operations and prove hypotheses.

- Benefit:
  - Proof reuse
Example

Challenge-Response Template

<table>
<thead>
<tr>
<th>A → B: m</th>
</tr>
</thead>
<tbody>
<tr>
<td>B → A: n, F(B,A,n,m)</td>
</tr>
<tr>
<td>A → B: G(A,B,n,m)</td>
</tr>
</tbody>
</table>

• Step 1:
  • Hypothesis: Function F(B,A,n,m) can be computed only by B
  • Property: Mutual authentication

• Step 2:
  • Instantiate F() to signature, keyed hash, encryption (ISO-9798-2,3, SKID3)
  • Satisfies hypothesis => Guarantees mutual authentication
Abstraction-Instantiation Method(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

\[
\begin{align*}
A \to B & : g^a, A \\
B \to A & : g^b, E_K \{\text{sig}_B \{g^a, g^b, A\}\} \\
A \to B & : E_K \{\text{sig}_A \{g^a, g^b, B\}\}
\end{align*}
\]

Two templates:

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

- **Authenticated Key Exchange:**
  - Template for JFKi, ISO-9798-3.
  - Template for JFKr, STS, IKE, IKEv2

- **Key Computation:**
  - Template for Diffie-Hellman, UM, MTI/A, MQV

- Combining these templates
Synthesis: STS-MQV
Outline

- Derivation System
- Compositional Logic
- Formalizing Composition
- Formalizing Refinements
- Conclusions and Future Work
Conclusions

- **Protocol Derivation System:**
  - Systematizes the practice of building protocols from standard sub-protocols. Useful for:
    - Modular protocol analysis
    - Underpinning protocol design principles and patterns
    - Organizing related protocols in taxonomies
    - Protocol synthesis

- **Protocol Logic:**
  - Correctness proofs follow derivation steps.
  - Rigorous treatment of composition, refinement.
Work in Progress

- **Derivation System:**
  - Development of taxonomies
  - Tool support based on *especs*

- **Protocol Logic:**
  - Formalization of transformations
  - Automation of proofs
Publications

  - Abstraction and Refinement in Protocol Derivation [CSFW04]
  - Secure Protocol Composition [MFPS03]
  - A Derivation System for Security Protocols and its Logical Formalization [CSFW03]
  - A Compositional Logic for proving Security Properties of Protocols [CSFW01,JCS03]
- C. Meadows, D. Pavlovic.
  - Deriving, Attacking and Defending the GDOI Protocol
- Web page:
  http://www.stanford.edu/~danupam/logic-derivation.html