Formal Derivation of Security Protocols

Anupam Datta Ante Derek
John C. Mitchell Dusko Pavlovic

Stanford University Kestrel Institute

HCSS April 15, 2004

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

$$A \rightarrow B: \quad g^a$$
$$B \rightarrow A: \quad g^b$$

- Shared secret (with someone)
  - A deduces:

$$Knows(Y, g^{ab}) \supset (Y = A) \vee Knows(Y,b)$$

- Authenticated
- Identity Protection
- DoS Protection

Component 2

Challenge-Response

A	$\rightarrow$	B:	m, A
B	$\rightarrow$	A:	n, sig_B {m, n, A}
A	$\rightarrow$	B:	sig_A {m, n, B}

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

Composition

$m := g^a$

$n := g^b$

ISO-9798-3

$A \rightarrow B$: g^a, A
$B \rightarrow A$: g^b, $sig_B\{g^a, g^b, A\}$
$A \rightarrow B$: $sig_A\{g^a, g^b, B\}$

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

Refinement

Encrypt Signatures

$A \rightarrow B$: g^a, A
$B \rightarrow A$: g^b, $E_K\{sig_B\{g^a, g^b, A\}\}$
$A \rightarrow B$: $E_K\{sig_A\{g^a, g^b, B\}\}$

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

Transformation

Use cookie: JFK core protocol

$A \to B$: g^a, A
$B \to A$: g^b, $hash_{KB}\{g^b, g^a\}$
$A \to B$: $g^a, g^b, E_K\{sig_A\{g^a, g^b, B\}\}$, $hash_{KB}\{g^b, g^a\}$
$B \to A$: $g^b, E_K\{sig_B\{g^a, g^b, A\}\}$

- 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,…

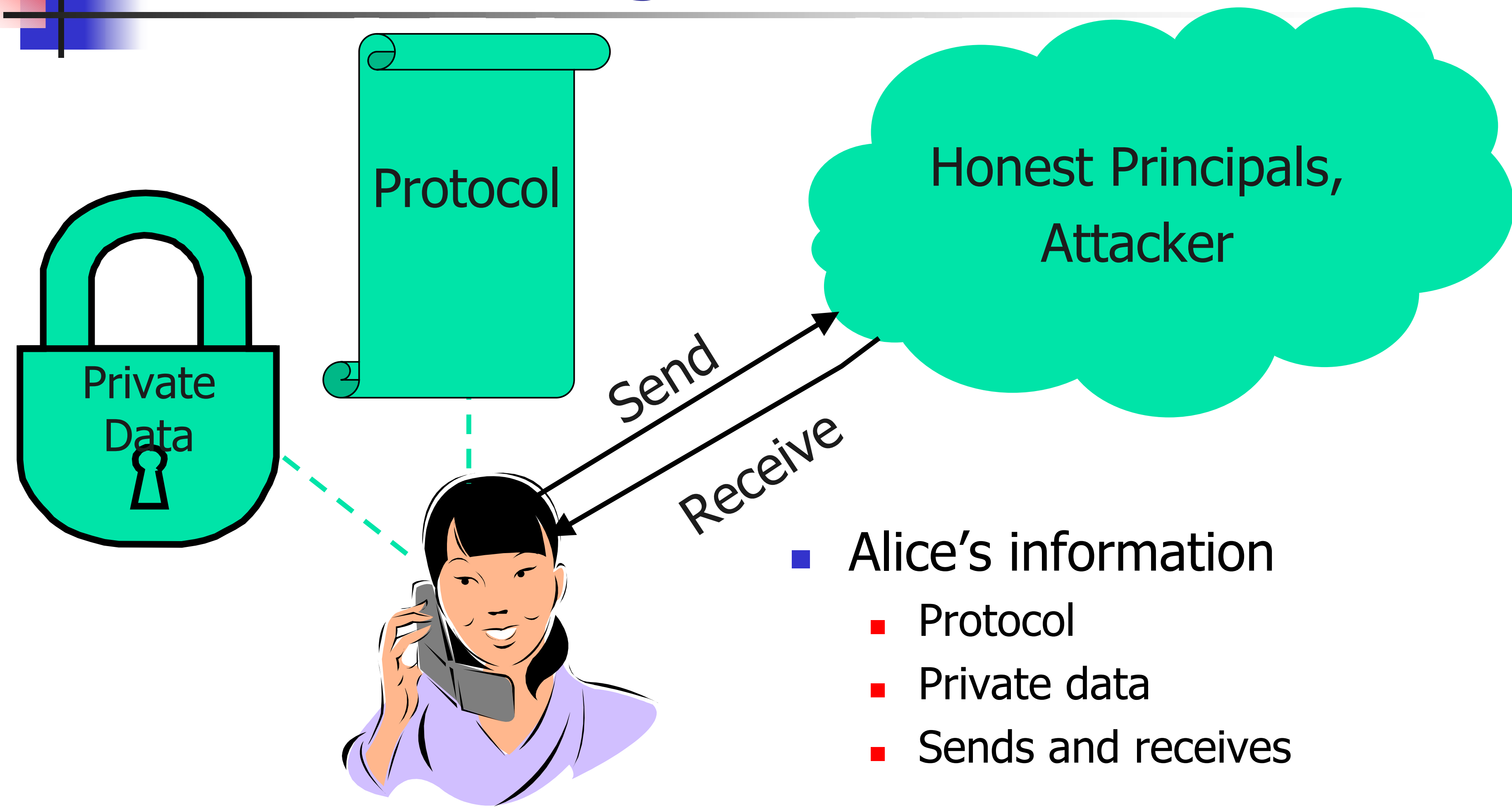
Benefits and Directions

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

Outline

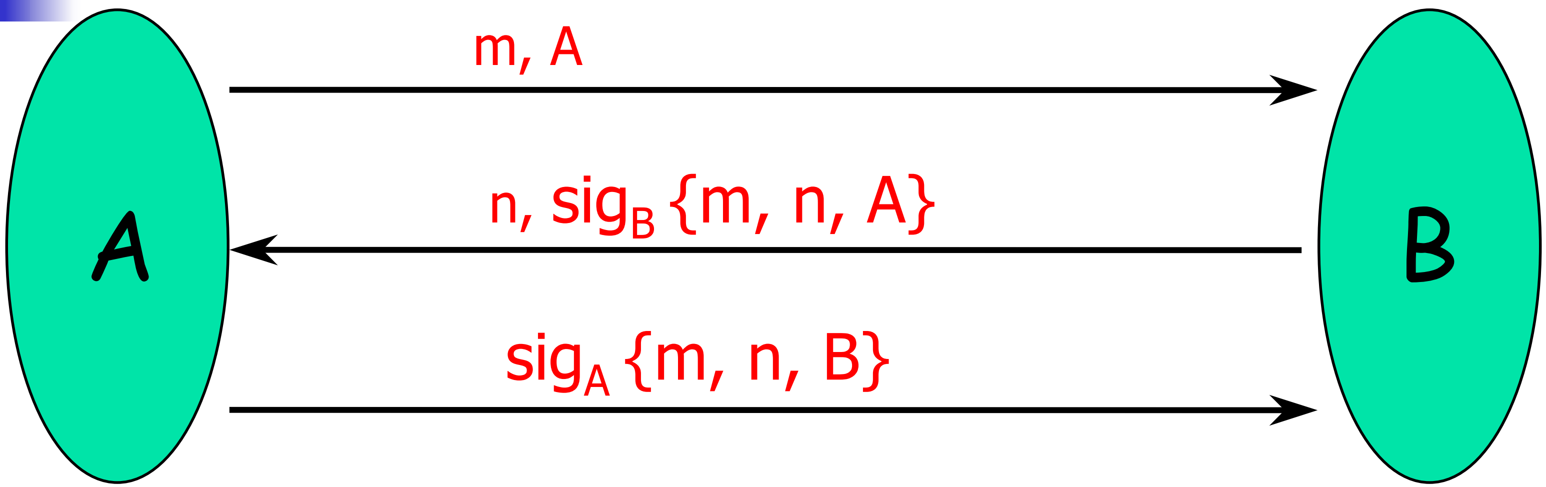
- Derivation System
- Compositional Logic [CSFW01,CSFW03]
  - Main idea
  - Syntax, semantics and proof system
- Formalizing Composition
- Formalizing Refinements
- Conclusions and Future Work

Protocol Logic: Main idea



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

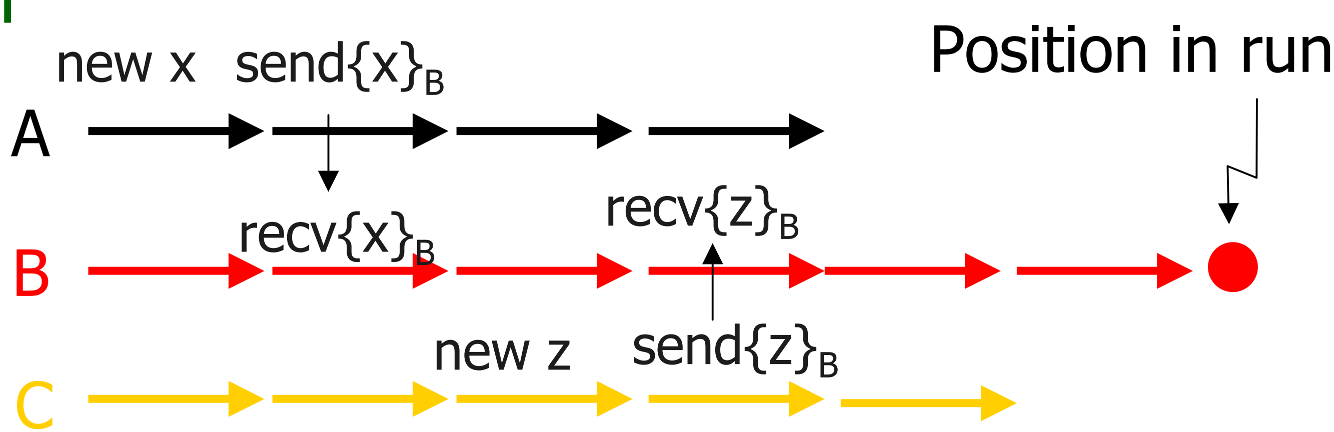
Example: Challenge-Response



- Alice reasons: if Bob is honest, then:
  - only Bob can generate his signature. [protocol independent]
  - if Bob generates a signature of the form 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, msg1) $\wedge$ Sent (B, msg2)

Execution Model

- Protocol
  - "Program" for each protocol role
- Initial configuration
  - Set of principals and key
  - Assignment of ≥1 role to each principal
- Run



Formulas true at a position in run

- Action formulas

  a ::= Send(P,m) | Receive (P,m) | New(P,t)

  | Decrypt (P,t) | Verify (P,t)

- Formulas

  φ ::= a | Has(P,t) | Fresh(P,t) | Honest(N)

  | Contains(t_1, t_2) | $\neg\varphi$ | $\varphi_1 \wedge \varphi_2$ | $\exists x\ \varphi$

  | $\circ\varphi$ | $\Diamond\varphi$

- Example

  After(a,b) = $\Diamond(b \wedge \circ\Diamond a)$

Modal Formulas

- After actions, postcondition

  $[\text{ actions }]_P\ \varphi$ where $P = \langle \text{princ, role id} \rangle$

  - If P does 'actions', starting from initial state, then φ holds in resulting state

- Before/after assertions

  $\varphi\ [\text{ actions }]_P\ \psi$

  - If φ holds in some state, and P does 'actions', then ψ holds in resulting state

Diffie-Hellman: Property

- Formula
  - $[\text{ new a }]_A$ 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: φ [actions]$_P$ ψ
  - precondition: Fresh(A,m)
  - actions: [Initiator role actions]$_A$
  - postcondition:

    Honest(B) $\supset$ ActionsInOrder(

    send(A, {A,B,m}),

    receive(B, {A,B,m}),

    send(B, {B,A,{n, sig$_B$ {m, n, A}}}),

    receive(A, {B,A,{n, sig$_B$ {m, n, A}}})

)

Proof System

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

    $\exists\ m' (\Diamond\text{Send}(X, m') \wedge \text{Contains}(m', \text{sig}_X\{m\})$
- 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
  - Has(A, {m,n}) $\supset$ Has(A, m) $\wedge$ Has(A, n)
  - Still good: unaffected by composition
- Protocol specific reasoning
  - "if honest Bob generates a signature of the form
    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 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

$$\frac{[\ ]_X\ \varphi \qquad \text{For all } B \in \text{BasicSeq}(Q).\ \varphi\ [B]_X\ \varphi}{Q \blacktriangleright \text{Honest}(X) \supset \varphi}$$

- Example

$$CR \blacktriangleright \text{Honest}(X) \supset (\text{Sent}(X, m_2) \supset \text{Recd}(X, m_1))$$

Composing protocols

Γ Γ'

DH ▶ Honest(X) $\supset$ …

Γ |- Secrecy

$\Gamma \cup \Gamma'$ |- Secrecy

Γ' |- Authentication

$\Gamma \cup \Gamma'$ |- Authentication

$\Gamma \cup \Gamma'$ |- Secrecy $\wedge$ Authentication [additive]

DH • CR ▶ $\Gamma \cup \Gamma'$ [nondestructive]

=

ISO ▶ Secrecy $\wedge$ Authentication

Composition Rules

- Invariant weakening rule

$$\frac{\Gamma \mid\!\!- \ \varphi\, [\ldots]_P\ \psi}{\Gamma \cup \Gamma' \mid\!\!- \ \varphi\, [\ldots]_P\ \psi}$$

- Sequential Composition

$$\frac{\Gamma \mid\!\!- \varphi\, [\, S\,]_P\, \psi \qquad \Gamma \mid\!\!- \psi\ [\, T\,]_P\, \theta}{\Gamma \mid\!\!- \varphi\, [\, ST\,]_P\, \theta}$$

- Prove invariants from protocol

$$\frac{Q \blacktriangleright \Gamma \qquad Q' \blacktriangleright \Gamma}{Q \bullet Q' \blacktriangleright \Gamma}$$

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

ISO-9798-2

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

SKID3

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

ISO-9798-3

$A \rightarrow B$: m
$B \rightarrow A$: n, $sig_B(n,m,A)$
$A \rightarrow B$: $sig_A(n,m,B)$

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

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

- 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

$A \rightarrow B$: g^a, A
$B \rightarrow A$: g^b, E_K {sig_B {g^a, g^b, A}}
$A \rightarrow B$: E_K {sig_A {g^a, g^b, B}}

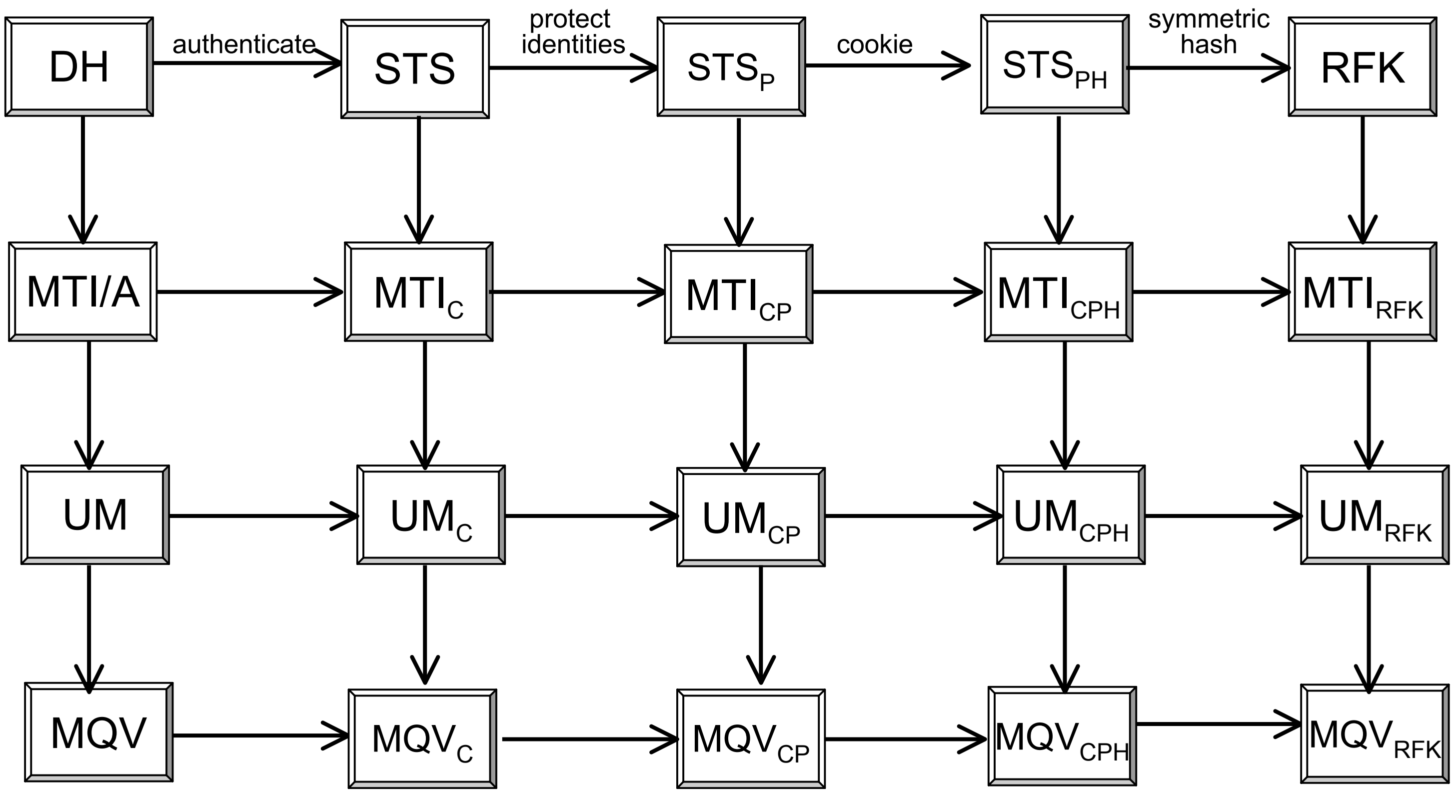
Two templates:

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

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

- A. Datta, A. Derek, J. C. Mitchell, D. Pavlovic.
  - Abstraction and Refinement in Protocol Derivation [CSFW04]
  - Secure Protocol Composition [MFPS03]
  - A Derivation System for Security Protocols and its Logical Formalization [CSFW03]
- N. Durgin, J. C. Mitchell, D. Pavlovic.
  - 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