Authentication Tests: Analyzing and Designing Cryptographic Protocols

Joshua D. Guttman
F. Javier Thayer
Jonathan C. Herzog
Lenore D. Zuck

March 2002

http://www.ccs.neu.edu/home/guttman/

Supported by the National Security Agency

Presented 21 March 2002

Clifford Lectures, Tulane University Mathematics Department

MITRE

Cryptographic Protocols

- For instance, Secure Sockets Layer (SSL)
  - Creates secure channel, browser to server
  - Agree on new shared secret
  - Use secret for encryption, integrity
- What is a cryptographic protocol?
  - Short, conventional sequence of messages
  - Uses cryptography
  - Goals: key distribution, authentication
- Frequently wrong
  - Even if the crypto is fine
  - May also amplify issues in crypto

MITRE

Trust Infrastructure

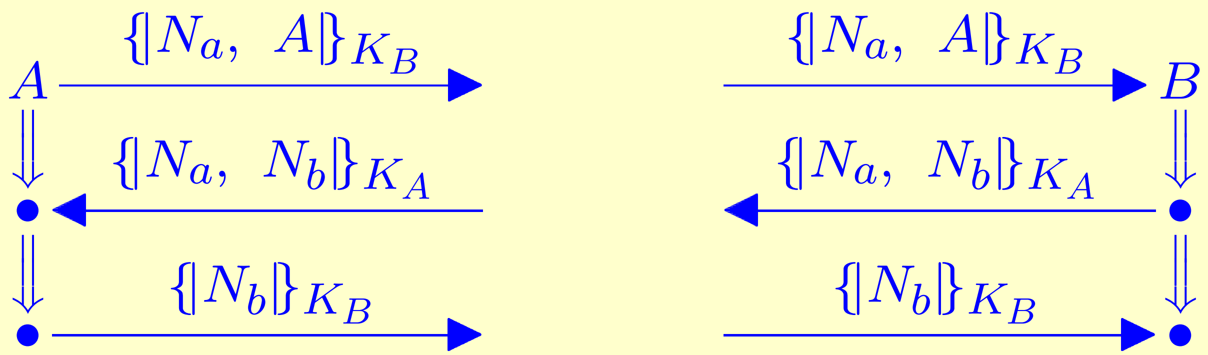
- Authenticate via cryptography
  - Principal demonstrates knowledge of
    - A private (asymmetric) key matching a certified public key, or
    - A shared secret key
  - Establishes identity
- Create new shared secrets
  - Entwined with authentication
  - Basis for secure conversation
  - Allows easy repeated authentication
- Preserve confidentiality or control access

Today's Goals

- Focus on one class of protocols, one type of flaw
  - Structural rather than cryptographic
- Explain how to prove correctness
- Illustrate how same ideas provide a protocol design method

MITRE

Example: Needham-Schroeder



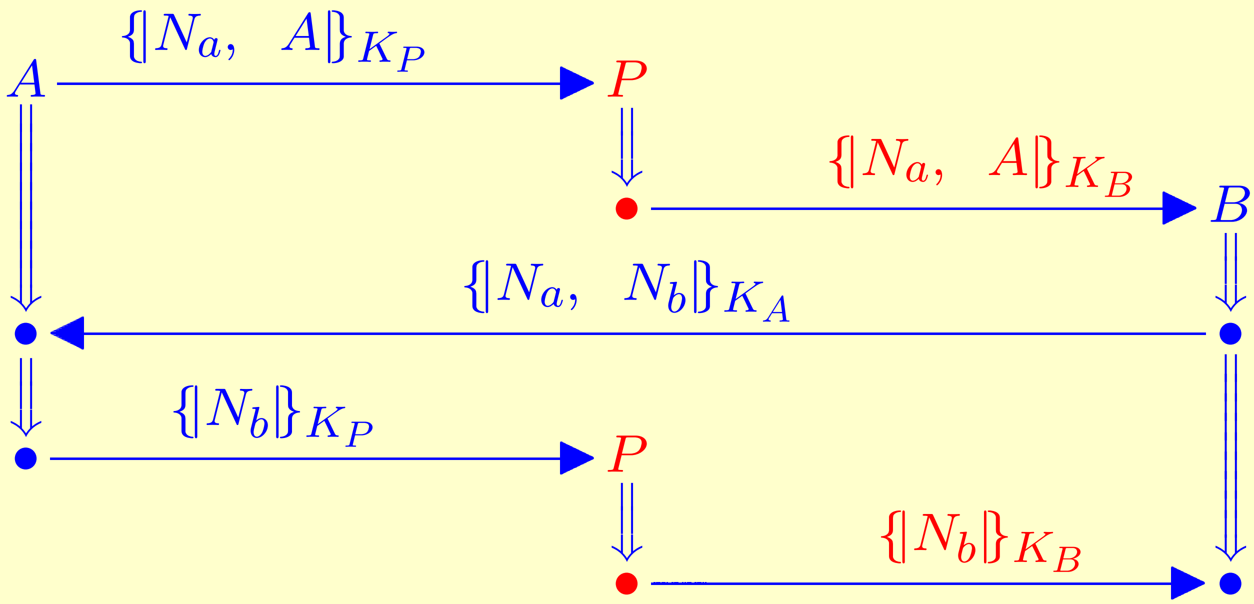
K_A, K_B	Public keys of A, B
N_a, N_b	Nonces, one-time random bitstrings
$\{\|t\|\}_K$	Encryption of t with K
$N_a \oplus N_b$	New shared secret

Why are Crypto Protocols Hard?

- Attacker chooses pattern of communication
- Attacker may also be a player
  - May hold keys
  - Will misuse them freely
- Attacker manipulates honest players
  - They play by the rules
  - Forced to serve as oracles
  - Protocol creates "unintended services"

MITRE

Needham-Schroeder Failure



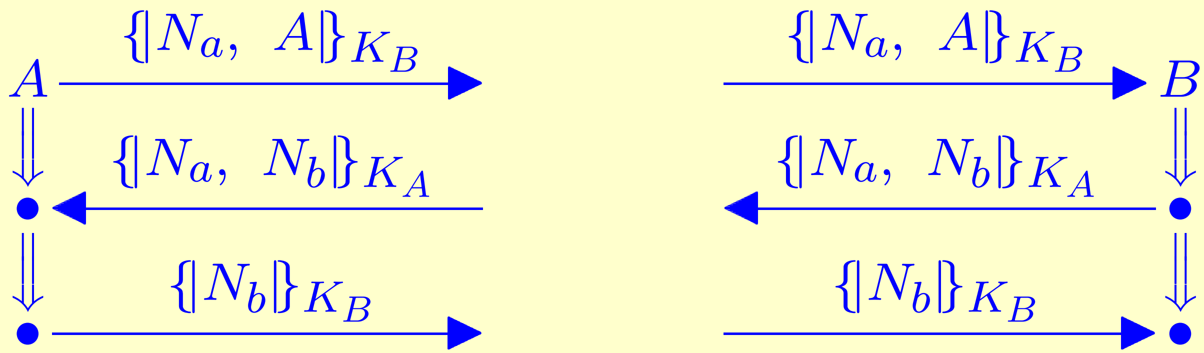
Due to Gavin Lowe, 1995

Diagnosis of a Failure

- Who was duped?
- Not A: Meant to share N_a, N_b with P
- B: Thinks he shares N_a, N_b only with A
  - Secrecy failed: P knows N_a, N_b
  - Authentication failed:
    - A had no run with B
    - B thinks A did

MITRE

Regular strands

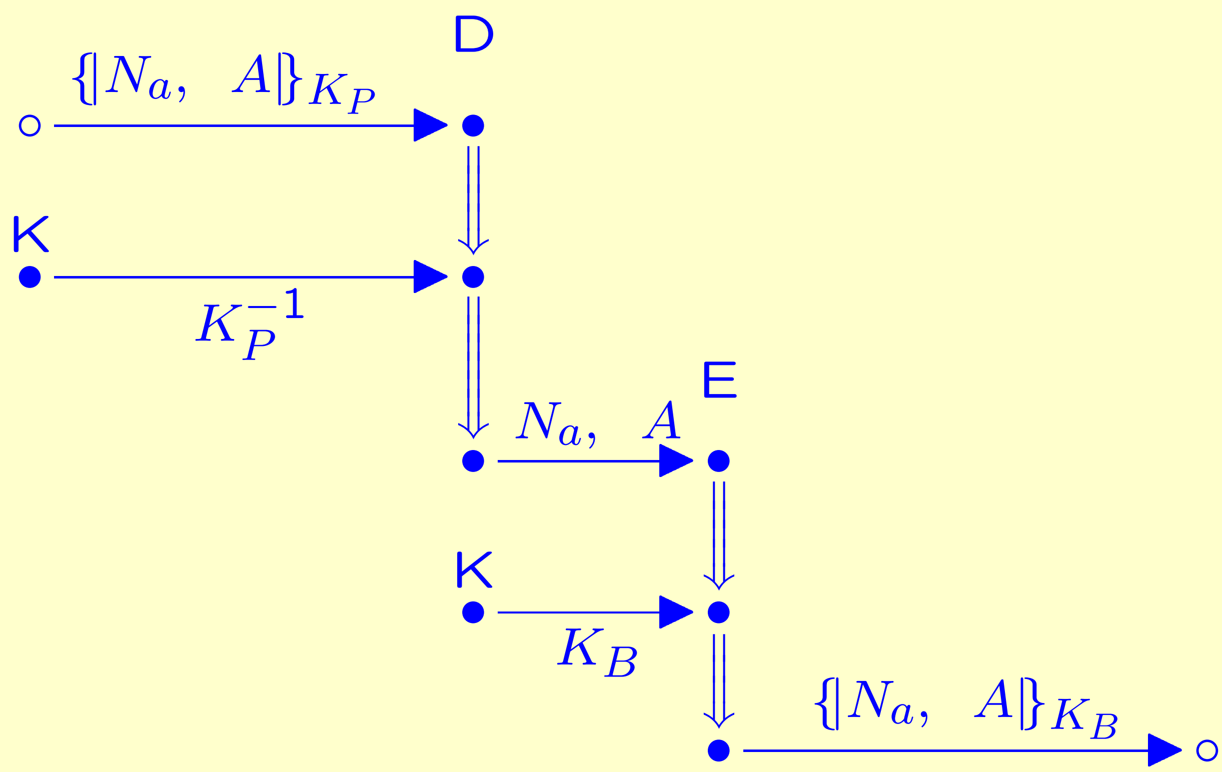


NSInit$[A, B, N_a, N_b]$

NSResp$[A, B, N_a, N_b]$

MITRE

NS Attack: Penetrator Activity



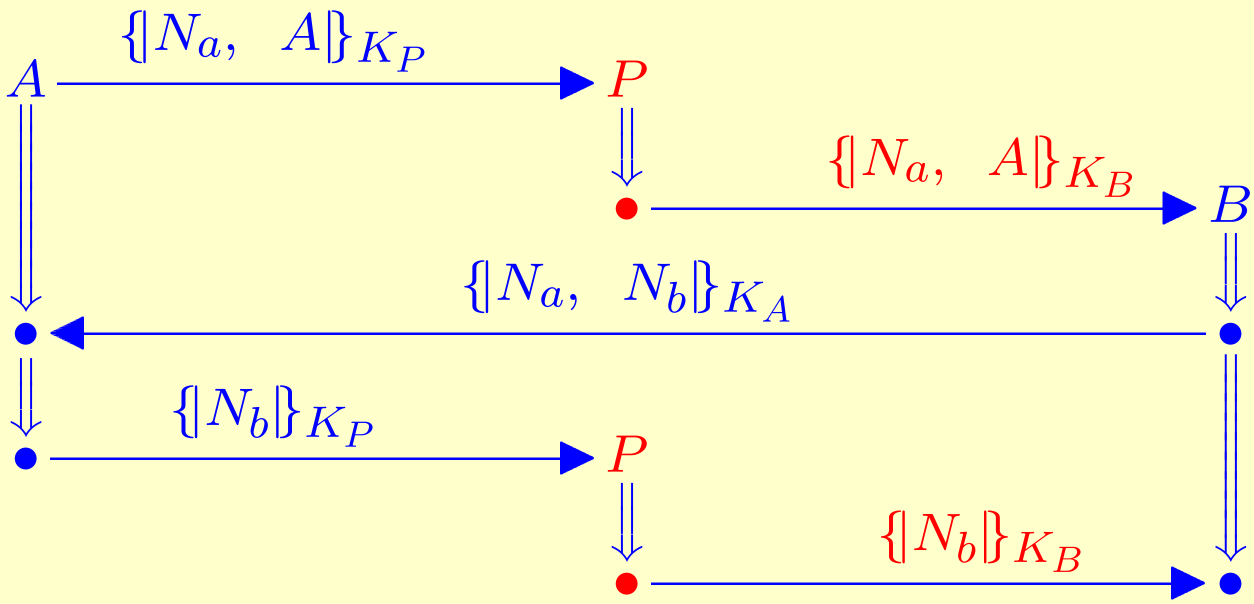
MITRE

Protocol Executions are Bundles

- Send, receive events on strands called “nodes”
  - Positive for send
  - Negative for receive
- Bundle $\mathcal{B}$: Finite graph of nodes and edges representing causally well-founded execution; Edges are arrows $\rightarrow$, $\Rightarrow$
  - For every reception $-t$ in $\mathcal{B}$, there's a unique transmission $+t$ where $+t \rightarrow -t$
  - When nodes $n_i \Rightarrow n_{i+1}$ on same strand, if n_{i+1} in $\mathcal{B}$, then n_i in $\mathcal{B}$
  - $\mathcal{B}$ is acyclic

MITRE

A Bundle



MITRE

Precedence within a Bundle

- Bundle precedence ordering $\preceq_{\mathcal{B}}$

  $n \preceq_{\mathcal{B}} n'$ means sequence of 0 or more arrows $\rightarrow$, $\Rightarrow$ lead from n to m

  $\preceq_{\mathcal{B}}$ is a partial order by acyclicity

  $\preceq_{\mathcal{B}}$ is well-founded by finiteness

- Bundle induction: Every non-empty subset of $\mathcal{B}$ has $\preceq_{\mathcal{B}}$-minimal members
- Reasoning about protocols combines
  - Bundle induction
  - Induction on message structure

MITRE

Messages

- Terms freely generated from
  - Names, texts
  - Nonces
  - Keys

  using the operators:
  - Concatenation $t_0,\ t_1$
  - Encryption with a key $\{\!|t_0|\!\}_K$
- Other algebras also interesting
  but today we'll use the free one

MITRE

Subterms and Origination

- Subterm relation $\sqsubset$
  least transitive, reflexive relation with

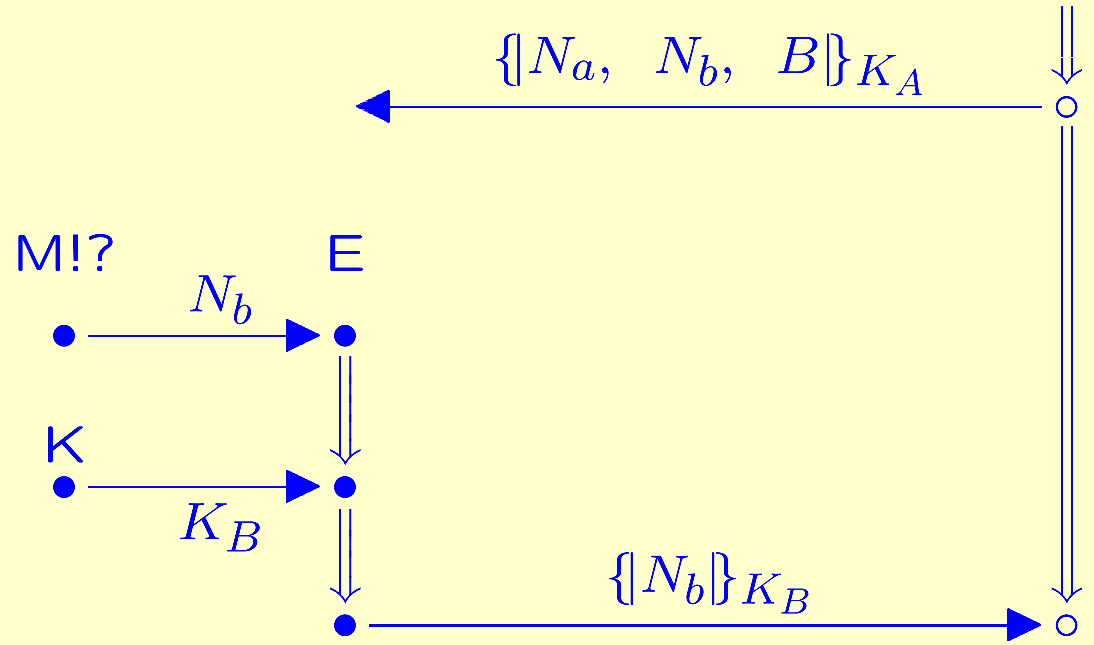
  $$g \sqsubset g,\ \ h$$
  $$h \sqsubset g,\ \ h$$
  $$h \sqsubset \{\!|h|\!\}_K$$

  N.B. $K \sqsubset \{\!|h|\!\}_K$ implies $K \sqsubset h$
- Represents *contents* of message, not how it's constructed
- t **originates** at n_1 means

  n_1 is a transmission $(+)$
  $t \sqsubset \text{term}(n_1)$
  if $n_0 \Rightarrow \cdots \Rightarrow n_1$, then $t \not\sqsubset \text{term}(n_0)$
- Unique origination, non-origination formalize
  a probabilistic assumption

MITRE

Guessing a Nonce



Guessing a private key (e.g. K_A^{-1})
similarly improbable

MITRE

An Authentication Goal

- Suppose:
  - Bundle $\mathcal{B}$ contains a strand Resp$[A, B, N_a, N_b]$
  - K_A^{-1} non-originating
  - N_b originates uniquely in $\mathcal{B}$
- Then:
  - There is a strand Init$[A, B, N_a, N_b]$ in $\mathcal{B}$

Authentication: correspondence assertions (of form $\forall\exists$)

This is false for NS

MITRE

A Secrecy Goal

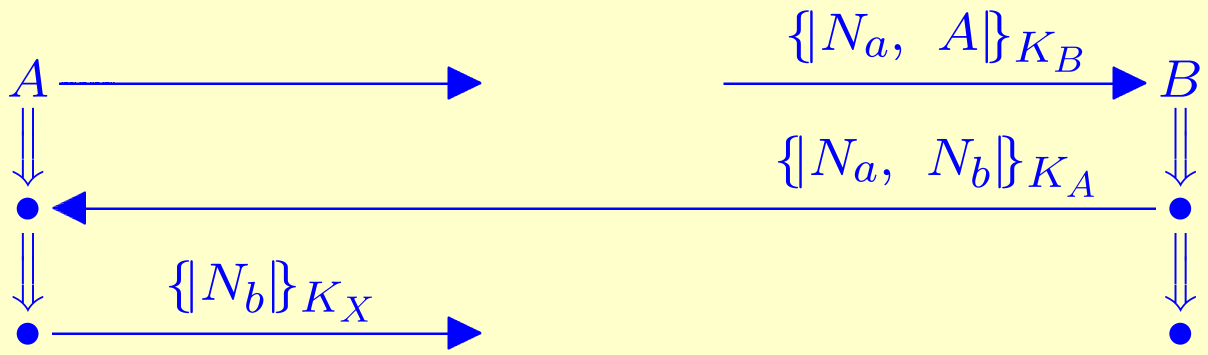
- Suppose:
  - Bundle $\mathcal{B}$ contains a strand Resp$[A, B, N_a, N_b]$
  - K_A^{-1}, K_B^{-1} non-originating
  - N_b originates uniquely in $\mathcal{B}$
- Then:
  - There is no node $n \in \mathcal{B}$ with term$(n) = N_b$

Form: $\forall$

This also is false for NS

MITRE

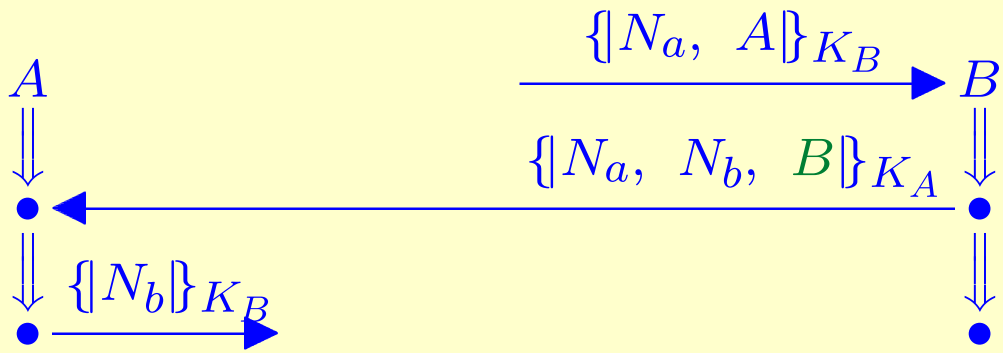
Why NS Fails



NSInit$[A, X, N_a, N_b]$ NSResp$[A, B, N_a, N_b]$

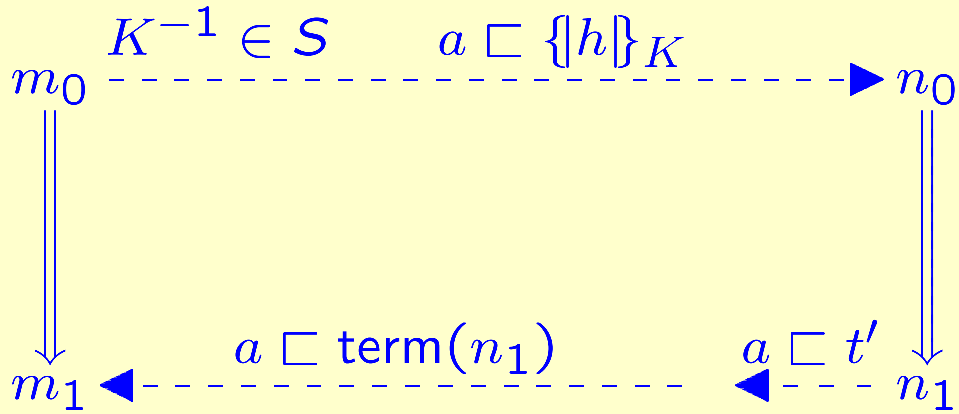
MITRE

Lowe's Fix



$\mathsf{NSInit}[A, B, N_a, N_b]$ $\mathsf{NSResp}[A, B, N_a, N_b]$

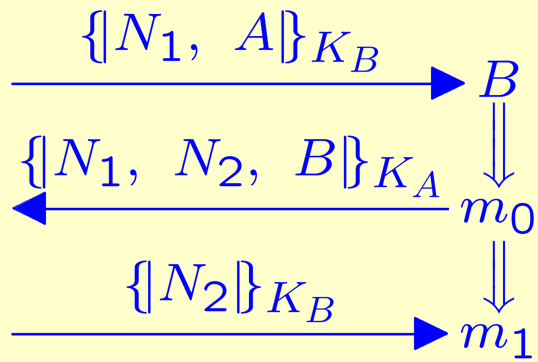
Outgoing Authentication Test



Assume $\{\!|h|\!\}_K \not\sqsubset \mathsf{term}(m_1)$
a originates uniquely at m_0,
a contained only in $\{\!|h|\!\}_K$

Conclude nodes n_0, n_1 exist in $\mathcal{B}$ and are regular
$\{\!|h|\!\}_K \not\sqsubset t'$
$m_0 \prec n_0 \prec n_1 \prec m_1$

NSL: Responder's Outgoing Test

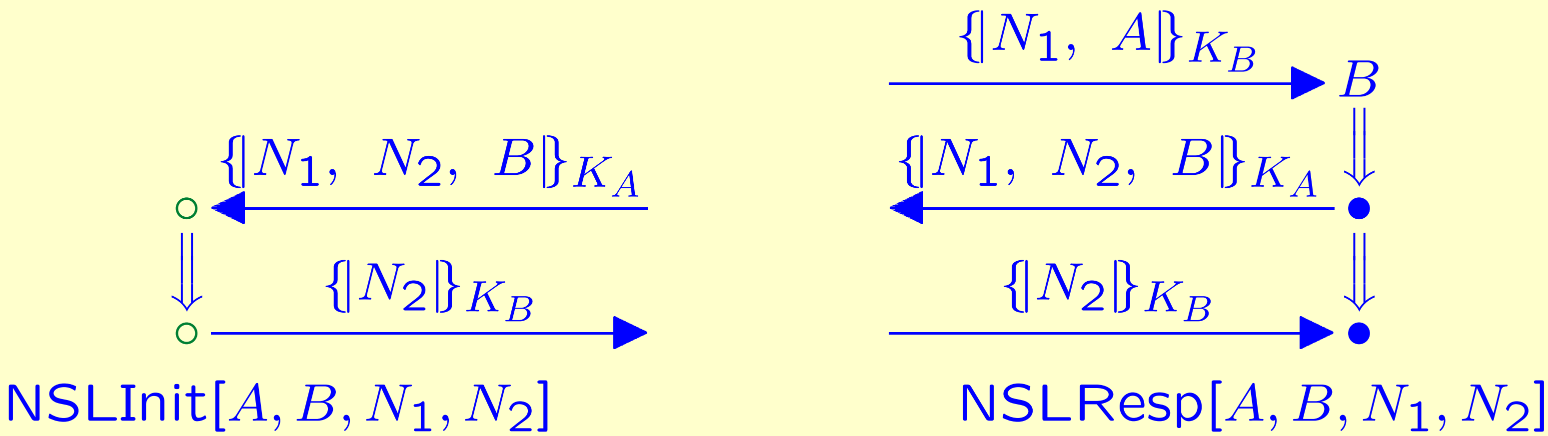


This is an outgoing test

What regular strand can transform $\{\!|N_1,\ N_2,\ B|\!\}_{K_A}$?

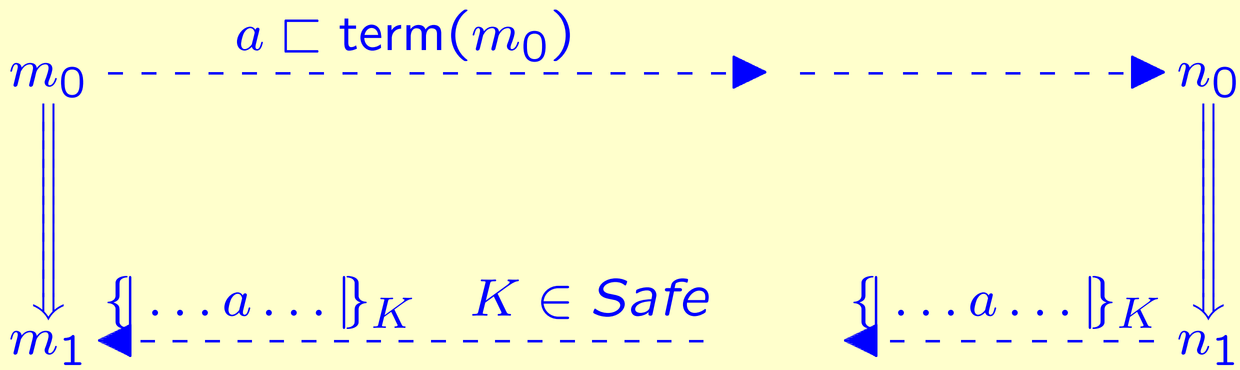
MITRE

Outgoing Test Conclusion



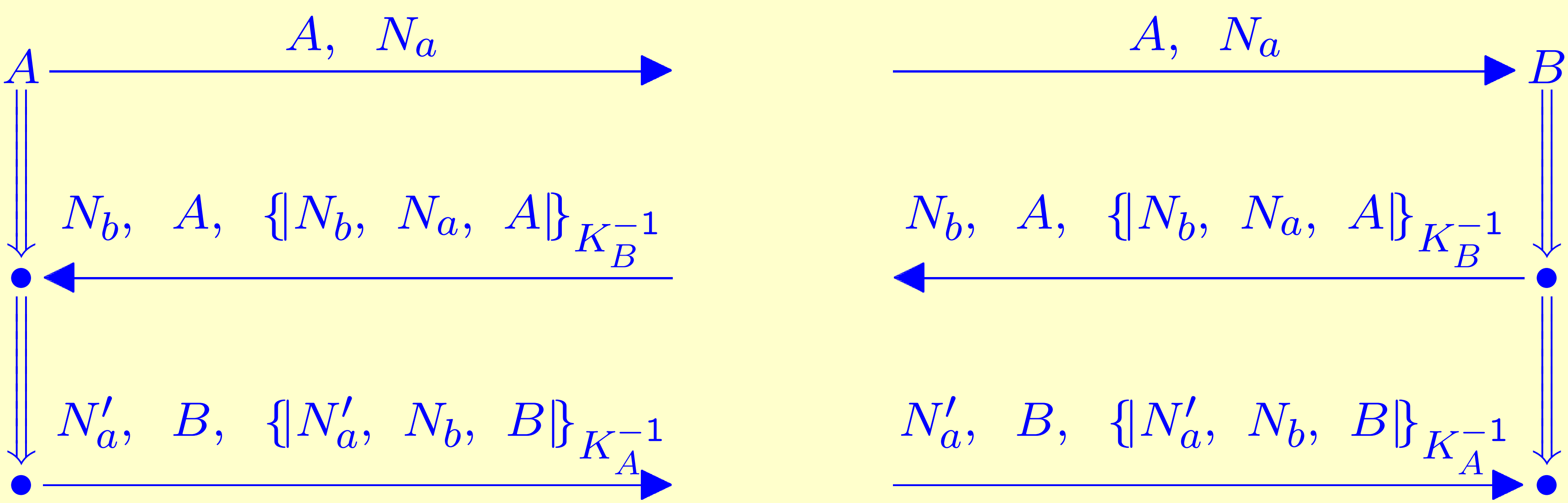
MITRE

Incoming Tests



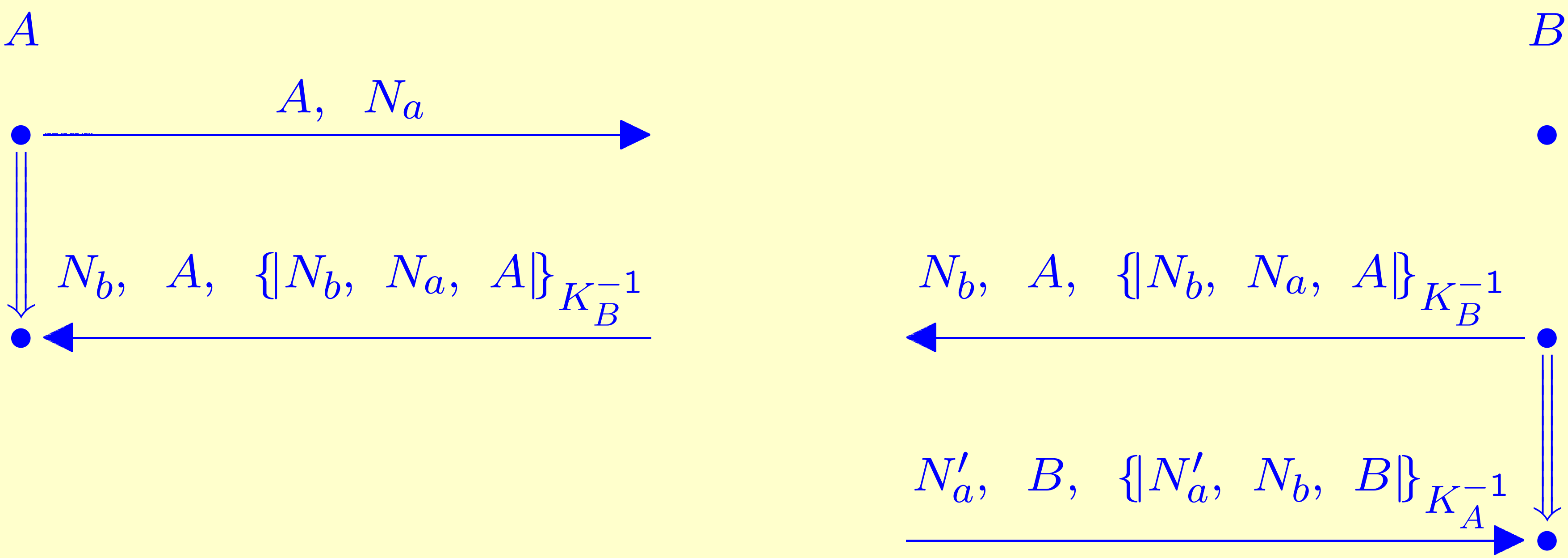
Assume	a originates uniquely at m_0
	$\{\!|\ldots a \ldots|\!\}_K \not\sqsubset \mathsf{term}(m_0)$
Conclude	nodes n_0, n_1 exist in $\mathcal{B}$ and are regular
	$m_0 \prec n_0 \prec n_1 \prec m_1$

Another Protocol (ISO reject)



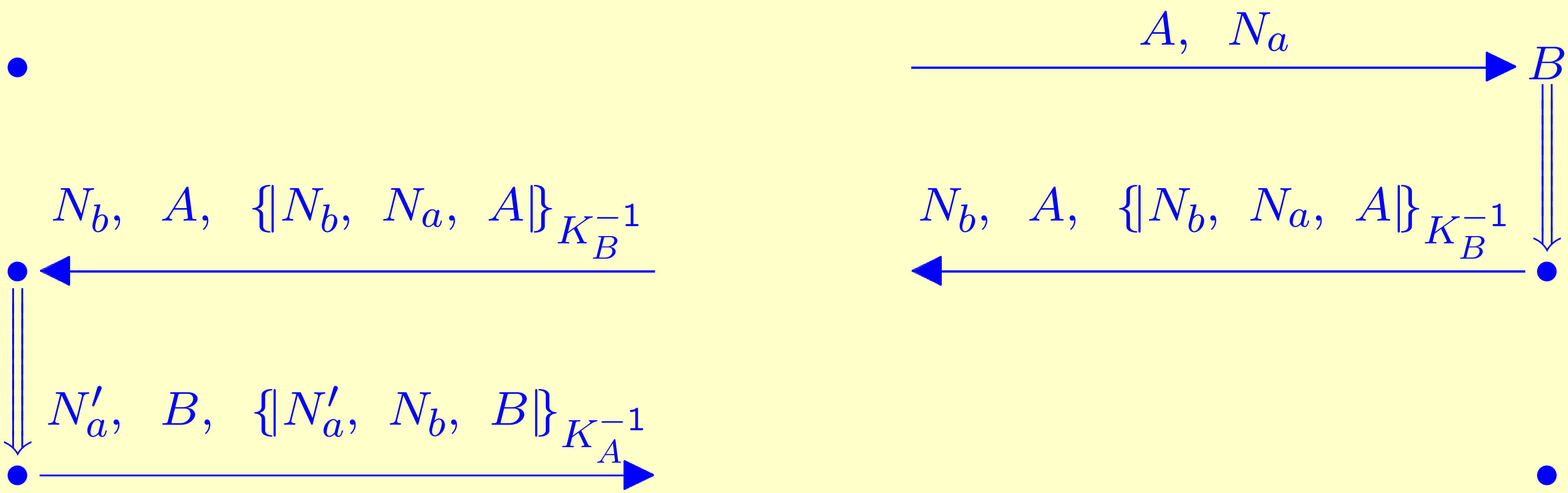
Mere authentication, using incoming tests

The Incoming Tests



MITRE

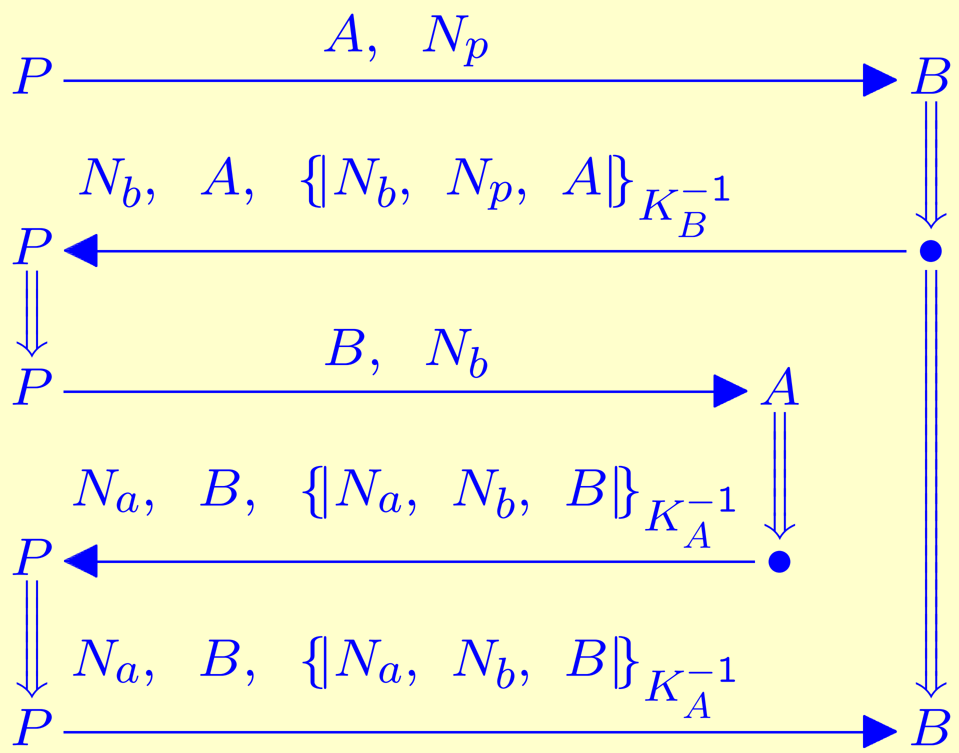
The Transforming Edges



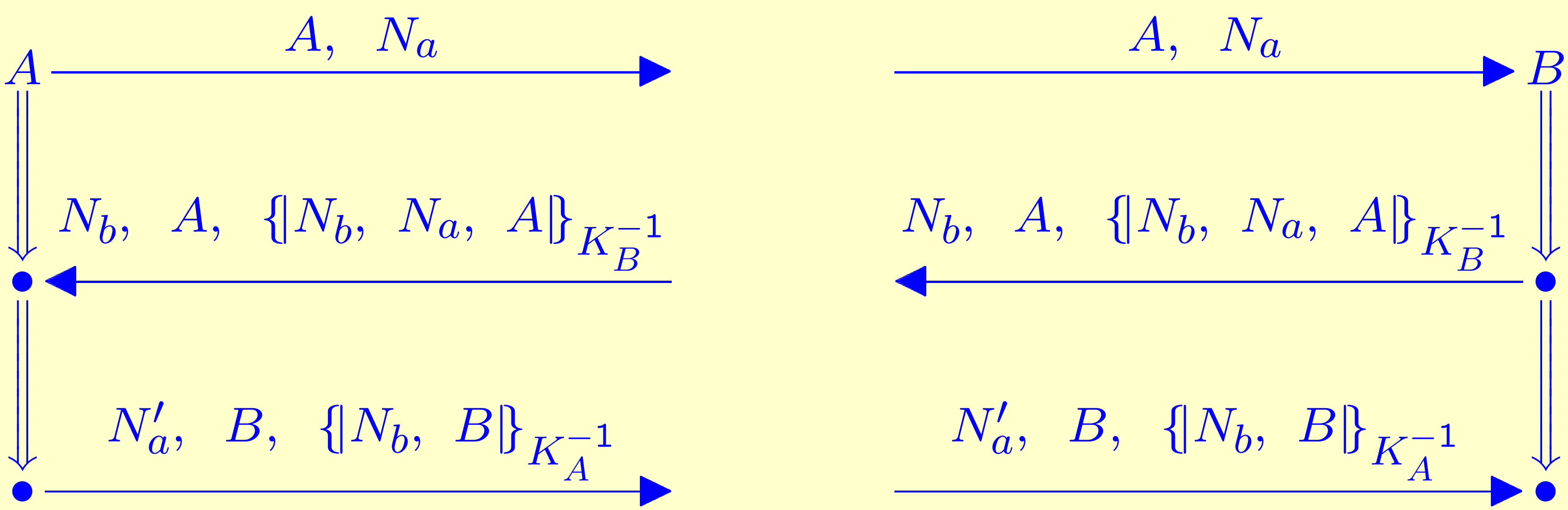
Produce same term
(just rename free variables)

MITRE

Counterexample to One Security Goal

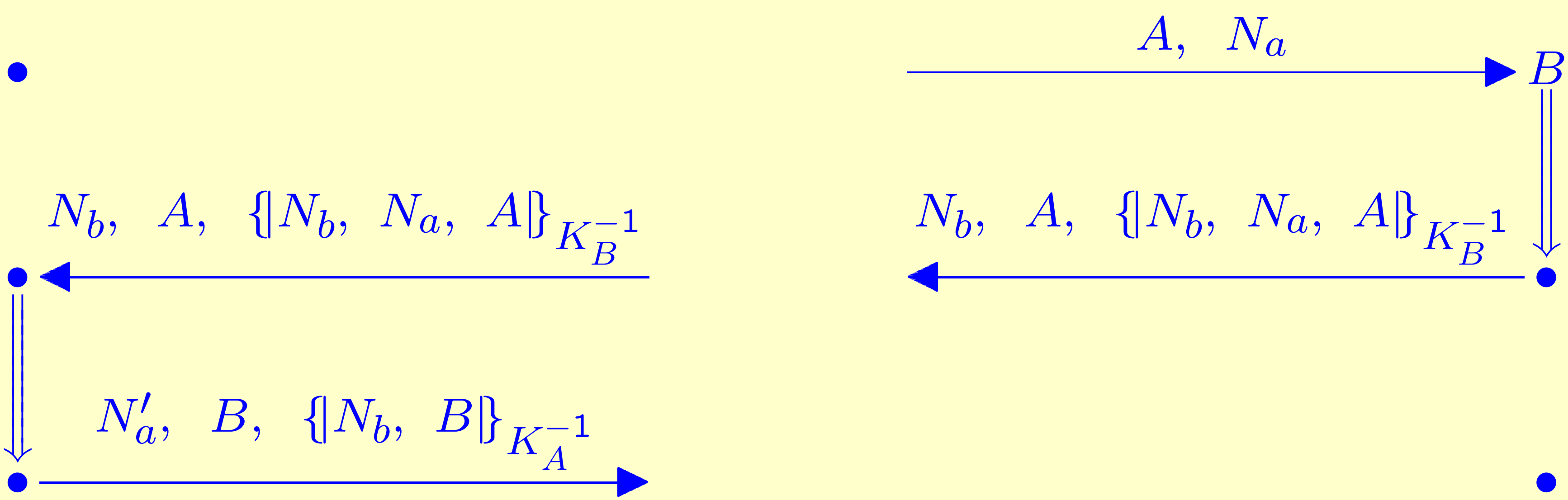


ISO Reject: Corrected Version



MITRE

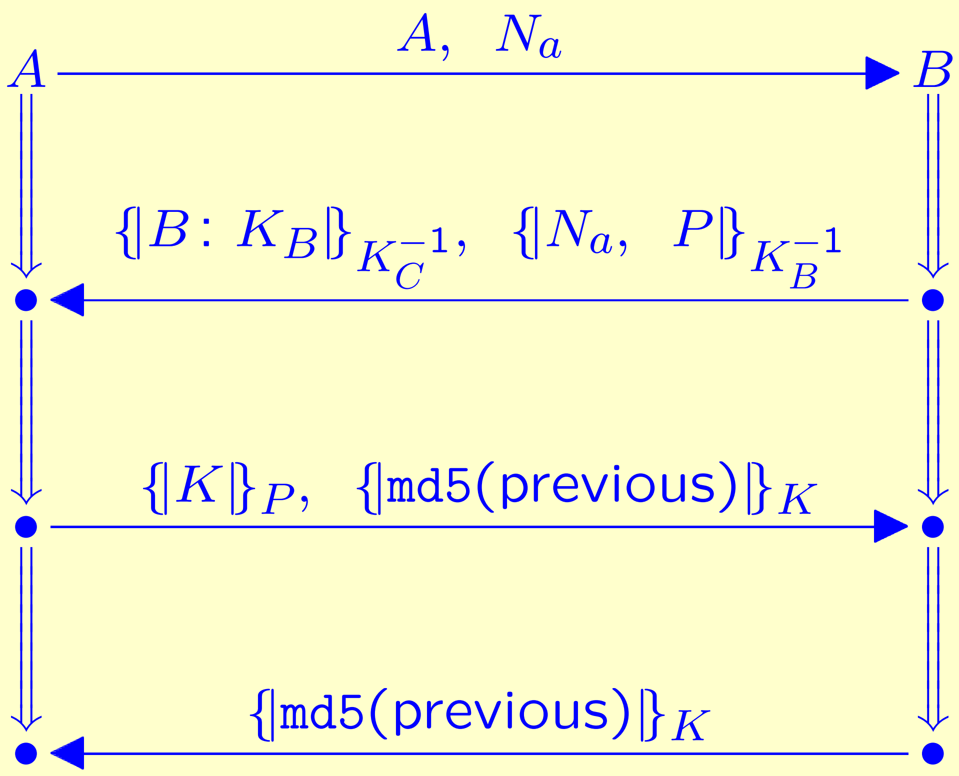
The Transforming Edges



Each test now requires a single, explicit transforming edge

MITRE

SSSL, a Simplified SSL



MITRE

Protocol Design

- Largely a matter of
  - selecting incoming, outgoing tests
  - inserting a single, explicit transforming edge for each
- Choosing an example: comparison with SSL
  - Provides good secrecy and authentication
  - Requires customer to trust merchant
  - Frequently undesirable
- Better: three-party protocol for customer, merchant, and bank
  - Credit card number goes to bank only
  - Item purchased shared with merchant only
  - All three must agree on price

MITRE

Secure Electronic Transaction

- SET protocol:
  - Visa, MasterCard, bank alliance
  - Protocol complete in 1997
  - In use nowhere
- Spectacularly complex
  - Hard to analyze
  - Hard to implement
  - Creates risk
- Our goal:
  simple, correct by design alternative

MITRE

Protocol Goals

Participants: Customer C, Merchant M, Bank B

Confidentiality All data to remain secret
Data for a pair not to be disclosed to third participant

Authentication, I Each P receives guarantee:
Q received and accepted P's data

Non-Repudiation P can prove its **Authentication, I** guarantee to a third party

Authentication, II Each Q receives guarantee:
data purportedly from P originated with P, in a recent run

MITRE

Assumptions

- Uncompromised public/private keypairs:
  - Private signature key
    (Public part for verification)
  - Private decryption key
    (Public part for encryption)

  We write $[\![h]\!]_P$, $\{\!| h |\!\}_P$
- Good hash function h

MITRE

Two Party Subprotocols

- Goals are essentially pair-wise
  (except confidentiality for shared data)
- Hence, design set of six two-party subprotocols
  - $C.M$, $C.B$, $M.B$, etc.
  - Each $P.Q$ achieves goals for role P
- Piece them together, later

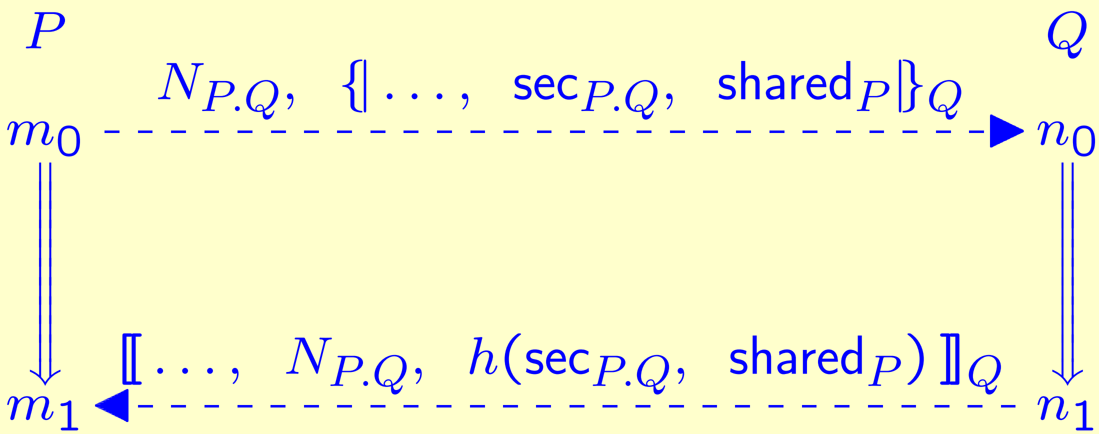
Confidentiality Send data as $\{\!|\ \ldots,\ \ \mathsf{sec}_{P.Q},\ \ \mathsf{shared}_P |\!\}_Q$

MITRE

Authentication, I

Each P receives guarantee: Q received and accepted P's data

- Use incoming test:



MITRE

Non-Repudiation

P can prove its **Authentication, I** guarantee to a third party

- No additional protocol contents needed
  - P discloses $N_{P.Q}, \ \ldots, \ \mathsf{sec}_{P.Q}, \ \mathsf{shared}_P$
  - Third party verifies signature

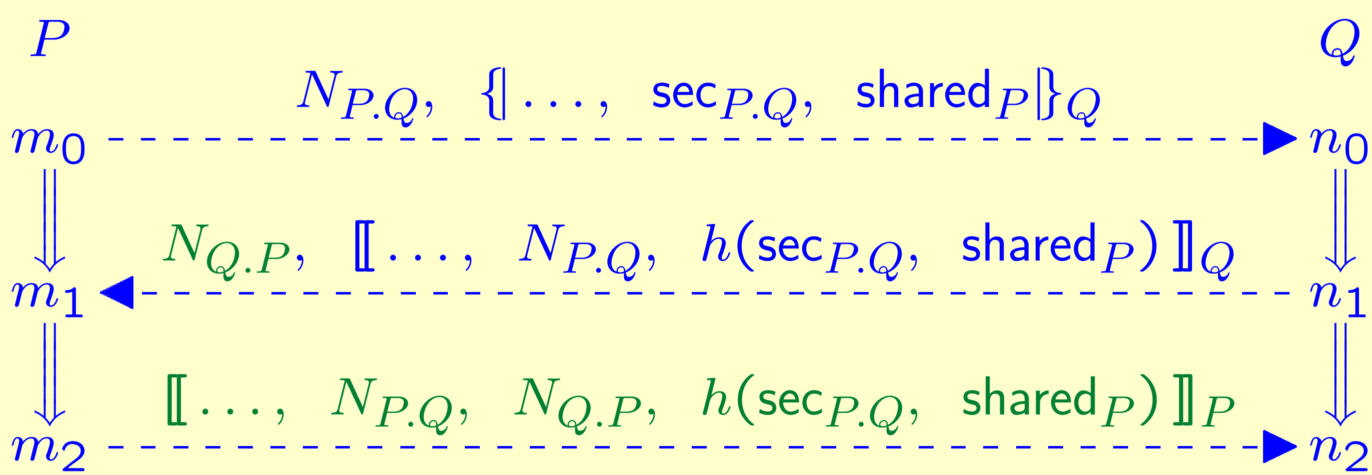
    $[\![\ldots, \ N_{P.Q}, \ h(\mathsf{sec}_{P.Q}, \ \mathsf{shared}_P)]\!]_Q$

MITRE

Authentication, II

Each Q receives guarantee:
data purportedly from P originated with P, in a recent run

- Again, use incoming test (right-to-left)



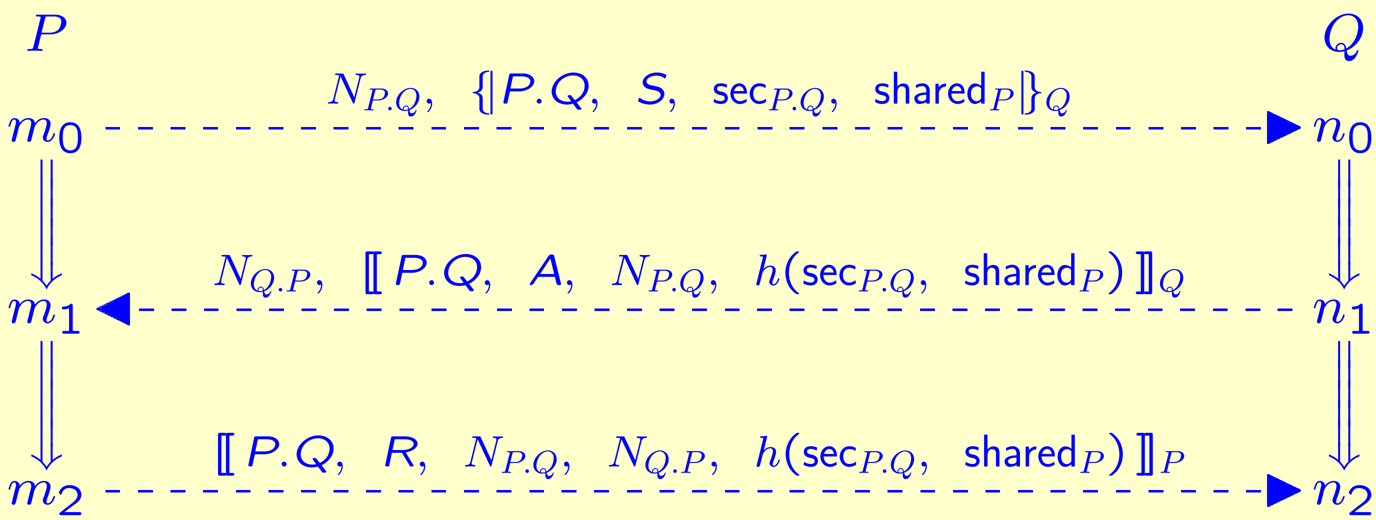
MITRE

Preventing Confusion among Subprotocols

- Multiple protocols on same network lead to failures
  - New transforming edges
  - Undermine authentication tests
- We have just designed six protocols
  - Are they still right if executed together?
  - Safer to tag each message with protocol name
    $C.M$, $C.B$, $M.B$, etc
- General theorem:
  disjoint encryption guarantees protocol independence
  (CSFW 2000)

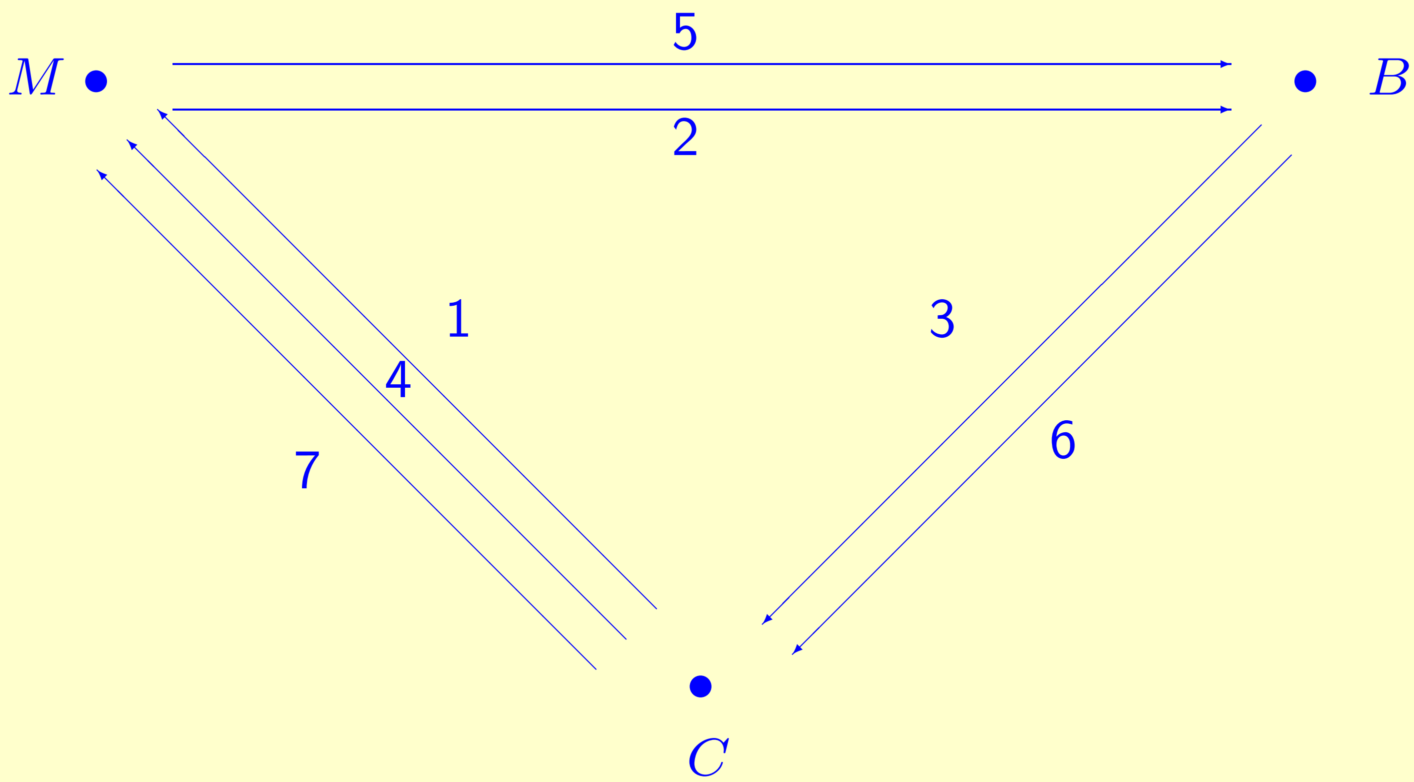
MITRE

Final Two-Party Protocol



MITRE

Piecing together the Three Party Protocol



Coordinating the subprotocols

- When to start:
  - C starts when ready
  - M starts on receipt of $C.M$ messages
  - B starts on receipt of $C.B$ messages
- When to emit new messages
  - On receipt of a $P.Q$ message
    P or Q follows the subprotocol
- When to forward message
  - On receipt of a $P.Q$ message
    forward it if neither P nor Q

MITRE

Protocol Design via Authentication Tests

- Designed new electronic commerce protocol
  - Trust relations in electronic transactions
  - Uniform, correct-by-design protocol
- Authentication tests:
  - Strong protocol proof method
  - Strong heuristic for design
- But:
  - Purely structural
  - Assume crypto perfect
  - Additional issues if crypto imperfect
- Cryptographic protocols:
  trust infrastructure for distributed systems

MITRE