Protocol Independence and Protocol Design

Joshua D. Guttman
F. Javier Thayer

September 2000

Protocol Independence

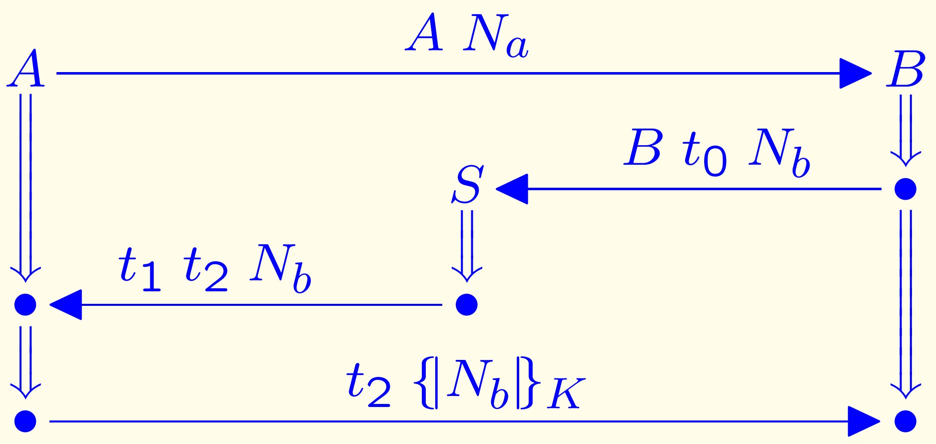
- Protocol independence problem
  - Protocols Π_1, Π_2 may be OK separately
  - But combination fails
- Protocol independence means

  If Π_1 meets security goal alone
  then Π_1 still does,
  in combination with Π_2
- Disjoint encryption for Π_1, Π_2
  - Π_2 never undoes encrypted terms created by Π_1
  - Π_2 never creates encrypted terms accepted by Π_1
- Disjoint encryption ensures protocol independence

The Problem: Mixing Protocols

- General informal advice: Avoid collisions
  - If keys always different, no problem
  - If each ciphertext incorporates a protocol number, no problem (but: be careful about session keys)
- Goal: Justify informal advice rigorously
  - Protocol independence: Protocols no worse in combination than separately
- Why mixing important
  - Potentially interfering protocols common:
    - Sub-protocols (e.g. TLS has 23)
    - Certificate management costs, re-use
    - Smart-card for several purposes
  - Technical interest: reasoning about multiple protocols

An Example: Neuman-Stubblebine, Part I

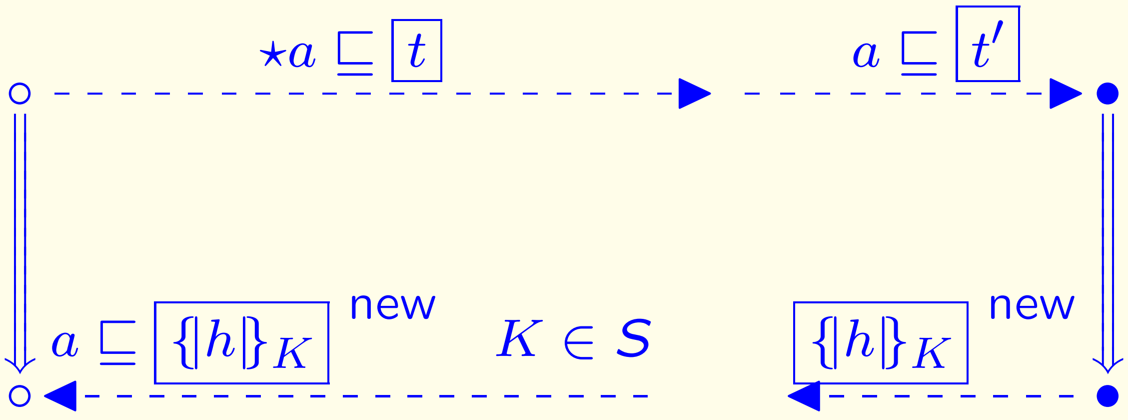


$t_1 = \{\!|B \; N_a \; K \; T|\!\}_{K_A}$ a "distribution"

$t_2 = \{\!|A \; K \; T|\!\}_{K_B}$ a "ticket"

$\{\!|N_b|\!\}_K$ a "confirmation"

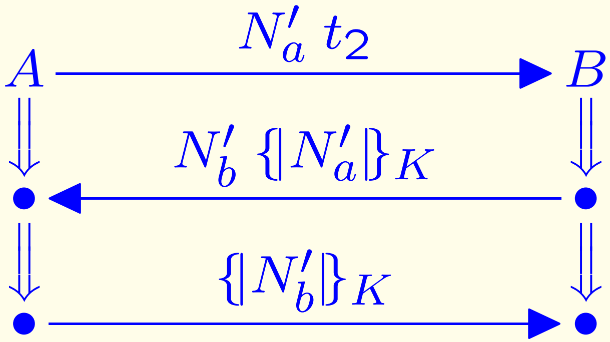
Incoming Test Authentication



A Goal: Responder's Guarantee

- Assume:
  - Server meets obligations
  - Long-term keys K_A, K_B uncompromised
  - Responder B has a complete strand, apparently with A
- Then:
  - There is a complete initiator strand with:
    - Same principals A, B
    - Same nonce N_b, timestamp T
    - Same session key K

Neuman-Stubblebine, Part II



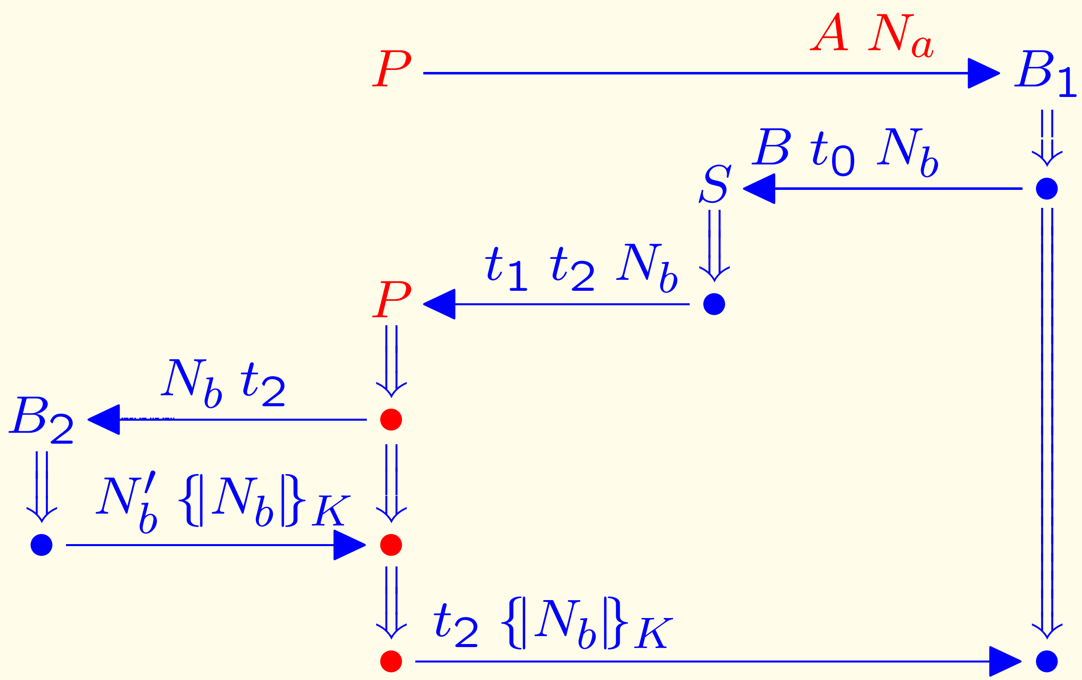
$t_2 = \{\!|A \; K \; T|\!\}_{K_B}$

Clearly, provides an unintended service:

$$N'_a \; t_2 \quad \Rightarrow \quad N'_b \; \{\!|N'_a|\!\}_K$$

So mixing causes attack on NS Part I

Attack on Mixed Neuman-Stubblebine



$t_2 = \{\!|A\ K\ T|\!\}_{K_B}$ a ticket

Main Ingredients in Attack

- Area of activity for each protocol

  Part I Strand B_1 and S

  Part II Strand B_2

- Connected by penetrator activity
  (point of view: Part I)

  Outbound Linking Paths From S to B_2

  Inbound Linking Paths From B_2 to B_1

- May assume bundle normal
  Each linking path has bridge term

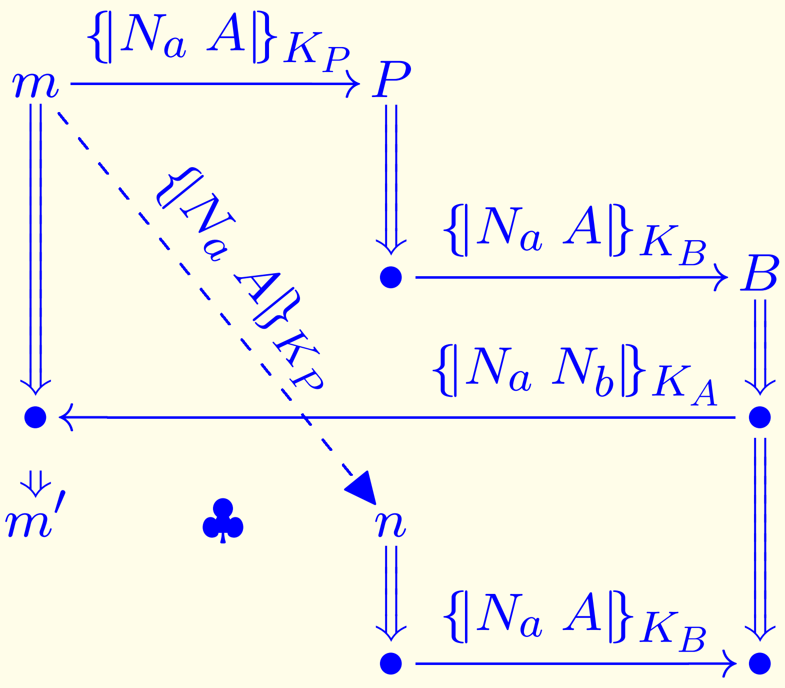
  Outbound N_b, t_2

  Inbound $\{\!|N_b|\!\}_K$

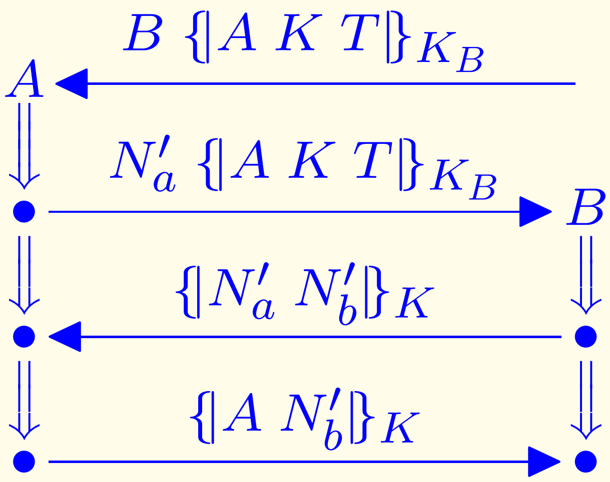
Inbound Bridge Terms

- Inbound bridge terms must be new components
  - Otherwise, make bundle efficient
  - Non-new inbound bridge terms gone
- For attacker,
  Part II is a generator for new components
  - Constructs terms accepted by Part I
  - Not available to penetrator via Part I
- Defender wants to destroy inbound bridges
  - Modify Part II to avoid new components accepted by Part I
  - Assures authentication goals preserved
- Secrecy goals: careful about outbound paths

An Efficient Bundle



Neuman-Stubblebine Part II, Corrected



First message fictitious:
Models state held by A
between run of part I and run of part II

- No new components accepted by Part I

Formalizing

- Multiprotocol strand space
  - $(\Sigma, tr), \Sigma_1$ where $\Sigma_1 \subset \Sigma$ and $s \in \Sigma$ implies s regular
- Σ_1 represents primary protocol

$$(\Sigma \setminus \Sigma_1) \setminus \mathcal{P} = \Sigma_2$$

  i.e. secondary protocol is non-primary regular

- Bundles $\mathcal{C}, \mathcal{C}'$ are equivalent iff they have the same primary nodes
  - Written $\mathcal{C} \equiv \mathcal{C}'$
  - Penetrator, secondary nodes may differ arbitrarily
- Protocol independence:

For every	$\mathcal{C}$
there exists	$\mathcal{C}'$ where $\mathcal{C} \equiv \mathcal{C}'$
	and $\mathcal{C}' \cap \Sigma_2 = \emptyset$

Equivalent Sub-Bundles

Suppose $\mathcal{C}$ a bundle and N a set of nodes.
Let G such that

1. $m \in G$
   if $m \in \mathcal{C}$ and
   $m \preceq_{\mathcal{C}} n$ for some $n \in N$
2. $m_1 \to m_2$
   if $m_1 \to m_2$ in $\mathcal{C}$
   and $m_1, m_2 \in G$
3. $m_1 \Rightarrow m_2$
   if $m_1 \Rightarrow m_2$ in $\mathcal{C}$
   and $m_1, m_2 \in G$

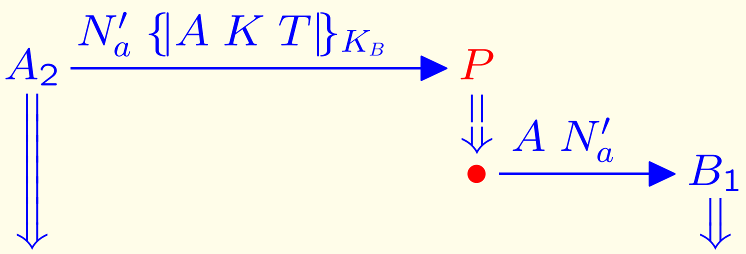
Then G is a bundle.

If $\mathcal{C} \cap \mathbf{\Sigma}_1 \subset N$ also, then $G \equiv \mathcal{C}$.

Strategy

- Define disjoint encryption, which restricts the encrypted components:
  - Sent by Σ_1 and received by Σ_2 (outbound)
  - Sent by Σ_2 and received by Σ_1 (inbound)
- Prove absence of inbound linking paths using efficiency
  - Equivalent Sub-Bundle result guarantees authentication goals met
- Ensure outbound linking paths disclose no secrets

Silly Counterexample



- Presumably N'_a originates uniquely on A_2
  - Can never get rid of that node without changing B_1
  - But origination of N'_a irrelevant to goals of primary protocol
- Security value:
  - Value potentially relevant to security goals of primary protocol

Catalog of Goal Ingredients

- Origination assumptions:
  - Uniquely originating values
  - Key server: session key originates uniquely
  - Non-originating values
- Authentication:

  If s_1 has $\mathcal{C}$-height i
  then s_2 has $\mathcal{C}$-height j

  where $s_1 \in \mathsf{Init}[\vec{v}]$,
  $s_2 \in \mathsf{Resp}[\vec{w}]$ (etc.)

  subject to origination assumptions on $\vec{v}, \vec{w}$
- Secrecy of v:
  - $v \not\sqsubseteq_{\emptyset} \mathsf{term}(n)$, for all $n \in \mathcal{C}$

    subject to origination assumptions...

What is a Security Value?

- Origination assumptions:
  constrain values used in primary protocol
  - Keys used on Σ_1, originating nowhere
  - Values originating uniquely on Σ_1
- Other values can occur anywhere
  - Values originating on Σ_2
  - Can also originate on penetrator strands
- Σ is full iff:

  If v originates on $s \in \Sigma_2$
  then v also originates on K or M strand
- Full spaces
  - Respect privacy values
  - Give penetrator other atomic values "free"

Disjoint Encryption

- Initial version (too crude):

$$\begin{array}{rl} \text{If} & n \in \Sigma_1 \text{ and } \{\!|h|\!\}_K \sqsubseteq \text{term}(n) \\ \text{and} & m \in \Sigma_2 \\ \text{then} & \{\!|h|\!\}_K \not\sqsubseteq \text{term}(m) \end{array}$$

- Initial version leaves out:
  - Emphasis on *new* components from Σ_2
  - Distinction between privacy values and others
- Disjoint outbound encryption:
  Let a private, $n_1 \in \Sigma_1$ pos., $n_2 \in \Sigma_2$ neg.

$$\begin{array}{rl} \text{Suppose} & a \sqsubseteq \{\!|h|\!\}_K \sqsubseteq \text{term}(n_1), \\ & \qquad \{\!|h|\!\}_K \sqsubseteq \text{term}(n_2) \\ \text{and} & n_2 \Rightarrow n_2' \\ \text{then} & a \not\sqsubseteq t \quad \text{if } \boxed{t}^{\text{new}} \sqsubseteq \text{term}(n_2') \end{array}$$

- Says Σ_2 doesn't re-package privacy values

No ZigZags

Let Σ have disjoint outbound encryption;
let $\mathcal{C}$ be well-behaved; let $(p, \mathcal{L})$ be a pedigree path for a

If $p_j \in \Sigma_1$
and $p_k \in \Sigma_2$ where $j < k$
then $a \neq \mathrm{term}(\ell(p))$

In particular, privacy values not disclosed via Σ_2

Disjoint Inbound Encryption

- Σ_2 doesn't make any new encrypted units accepted by Σ_1
- Def: Let $n_1 \in \Sigma_1$ neg., $n_2 \in \Sigma_2$ pos.

$$\begin{array}{rl} \text{If} & \{\!|h|\!\}_K \sqsubseteq \text{term}(n_1) \text{ and } \{\!|h|\!\}_K \sqsubseteq \text{term}(n_2) \\ \text{and} & \boxed{t_0}^{\text{new}} \sqsubseteq \text{term}(n_2) \\ \text{then} & \{\!|h|\!\}_K \not\sqsubseteq t_0 \end{array}$$

- Example: NS Part II vs. modified version