Sessions and Separability

in Security Protocols

Marco Carbone
Joshua D. Guttman

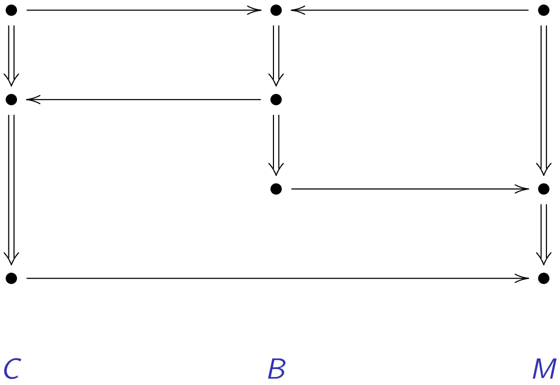
IT University of Copenhagen
Worcester Polytechnic Institute

Thanks to: Danish Agency for Science, Technology and Innovation
US National Science Foundation
(Grants CNS-0952287, CNS-1116557).

8 March 2013
BiSS

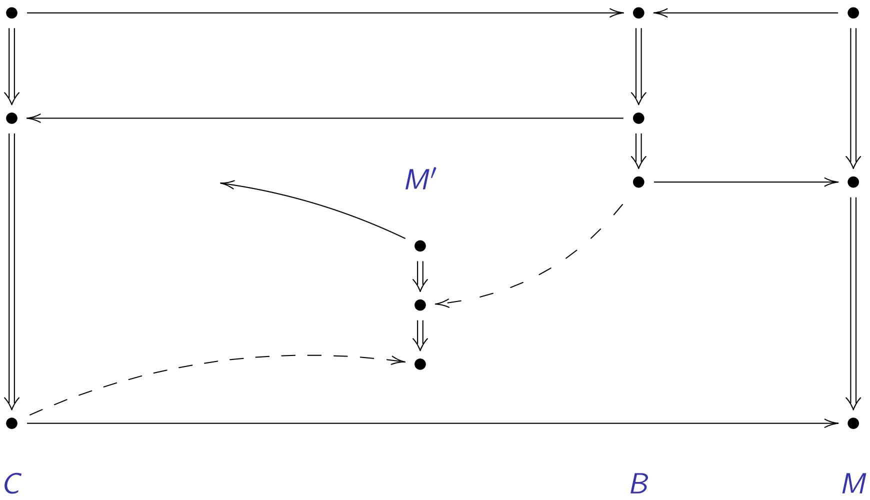
Why session behavior counts

Customer C buys from broker B with commission from manufacturer M

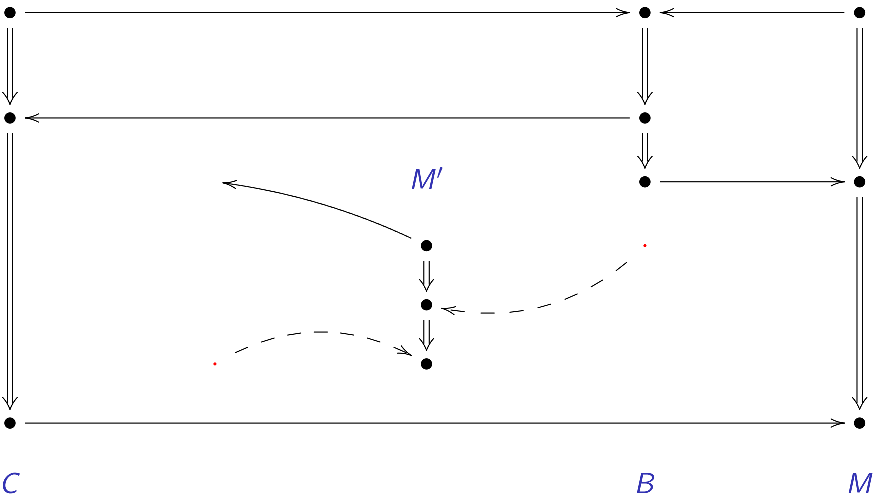


Getting paid twice

Can the broker get paid twice?



Separating M'



Session Behavior, v. 1

Relates local runs of participants to global session

A protocol Π has session behavior if, in every execution:

- Any two complete local runs that interact belong to the same global session — homogeneity
- Any two complete local runs of the same role belong to different global sessions — exclusion

Even with active, malicious adversary

Session Behavior, v. 1

Relates local runs of participants to global session

A protocol Π has session behavior if, in every execution:

- Any two complete local runs that interact belong to the same global session — homogeneity
- Any two complete local runs of the same role belong to different global sessions — exclusion

Even with active, malicious adversary

Whatever the adversary can do,
he can do without confusing sessions

Goals of this hour

- Define session behavior
- Give syntactic criteria for a protocol to have session behavior
- Develop proof methods "Separability theorem"

Goals of this hour

- Define session behavior
- Give syntactic criteria for a protocol to have session behavior
- Develop proof methods "Separability theorem"
- Free bonus:

Protocol independence results also follow from separability theorem

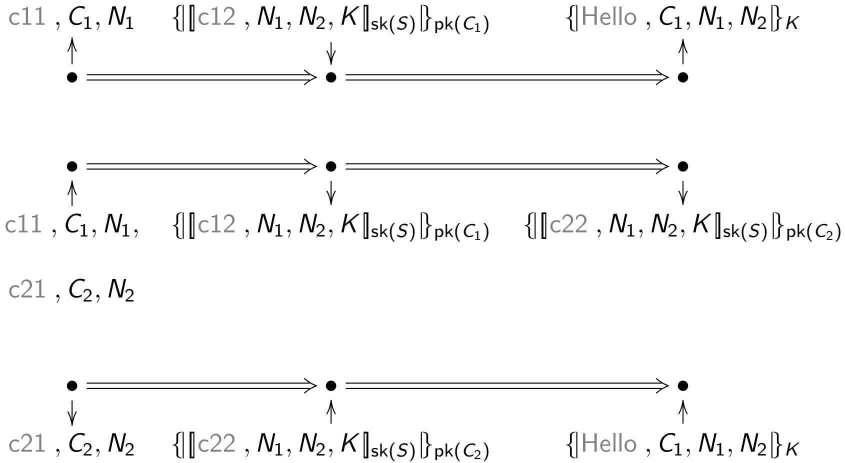
Exclusion principle requires freshness

- Each role ρ selects a fresh value a_ρ
- a_ρ called ρ's proper session parameter
- Distinct strands of same role
  - Choose different values for a_ρ, so
  - Belong to different global sessions
- Identify global session using session parameters a_ρ for the various roles
- Homogeneity principle requires:

  Crypto units should identify their session

A Session-respecting protocol

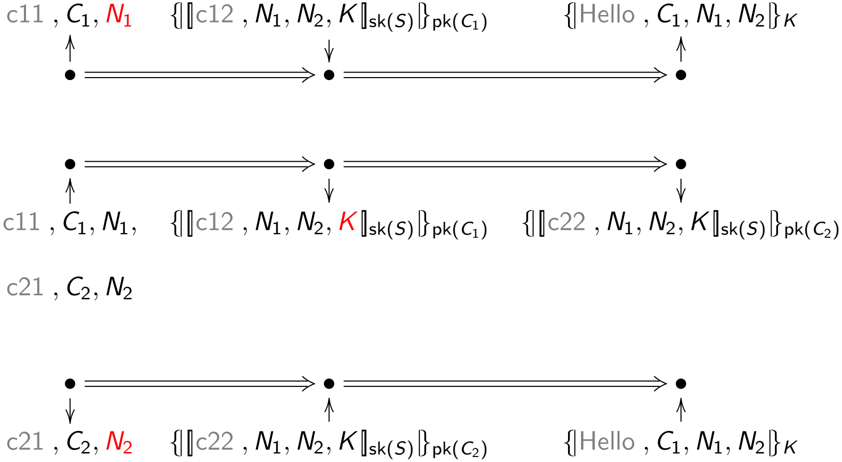
Brokered invitation — Roles: Host, broker, guest



A Session-respecting protocol

Brokered invitation

Roles: Host, broker, guest



Session parameters N_1, N_2, K in red

So why is this hard?

- Syntactic flexibility

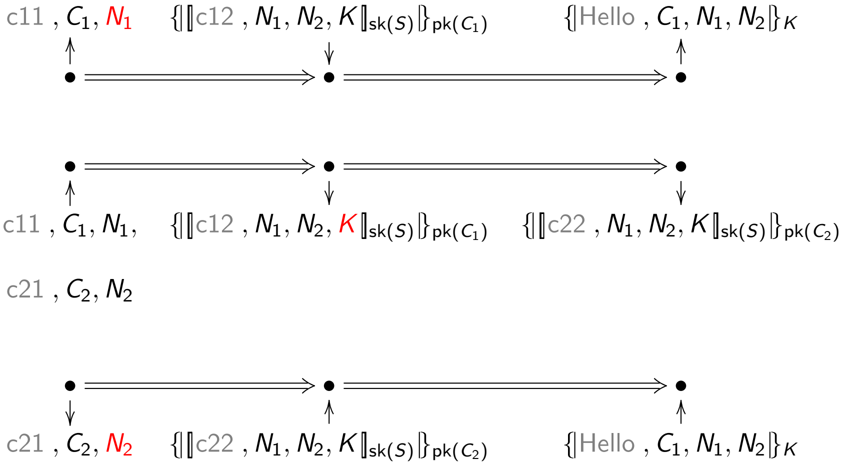
- Adversary model: Partial compromise

- “Same session” is not an equivalence relation

A Session-respecting protocol

Brokered invitation

Roles: Host, broker, guest



So why is this hard?

- Syntactic flexibility
  - Session parameters may contribute to plaintext or key
  - Participants may join late
- Adversary model: Partial compromise
- "Same session" is not an equivalence relation

Achieving session behavior

Contributive: Each encryption involves all session parameters acquired so far

Achieving session behavior

Contributive: Each encryption involves all session parameters acquired so far

No ambiguity: Two encryptions that unify agree on session parameters

Achieving session behavior

Contributive: Each encryption involves all session parameters acquired so far

No ambiguity: Two encryptions that unify agree on session parameters

Early arrivals x are acquired on first reception or first transmission

Achieving session behavior

Contributive: Each encryption involves all session parameters acquired so far

No ambiguity: Two encryptions that unify agree on session parameters

Early arrivals x are acquired on first reception or first transmission

Late arrivals y are acquired in encrypted form;
fresh values acquired on same node are encrypted

So why is this hard?

- Syntactic flexibility
  - Session parameters may contribute to plaintext or key
  - Participants may join late
- Adversary model: Partial compromise
  - Compliant participants get session behavior, despite compromised participants in session

double-commission problem

- "Same session" is not an equivalence relation

So why is this hard?

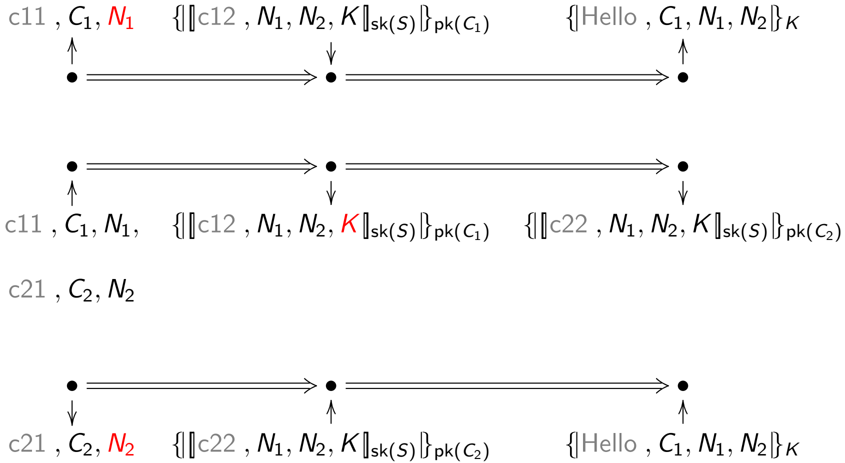
- Syntactic flexibility
  - Session parameters may contribute to plaintext or key
  - Participants may join late
- Adversary model: Partial compromise
  - Compliant participants get session behavior, despite compromised participants in session

double-commission problem

- "Same session" is not an equivalence relation
  - "May influence" is a preorder on nodes

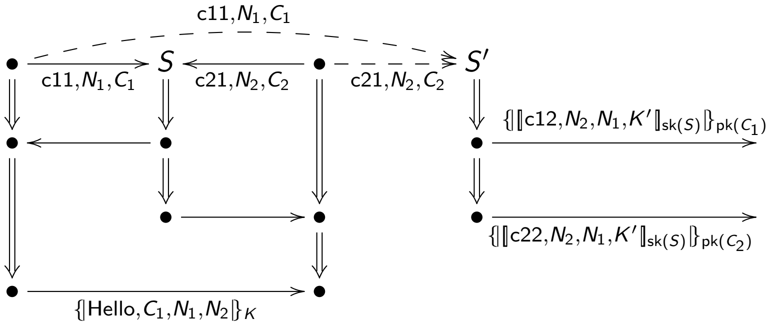
A Session-respecting protocol

Brokered invitation Roles: Host, broker, guest



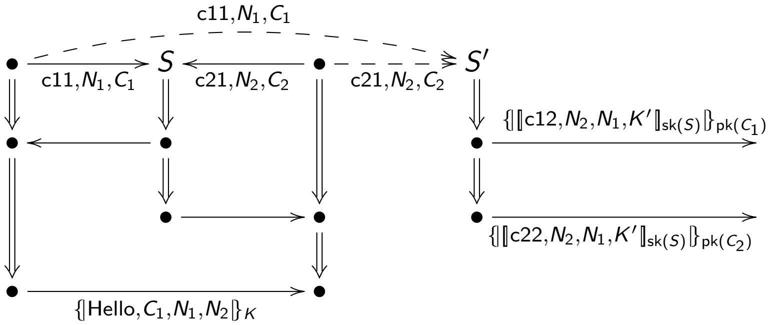
What about this? Execution $\mathcal{B}$

"Bundle" formalizes execution



What about this? Execution $\mathcal{B}$

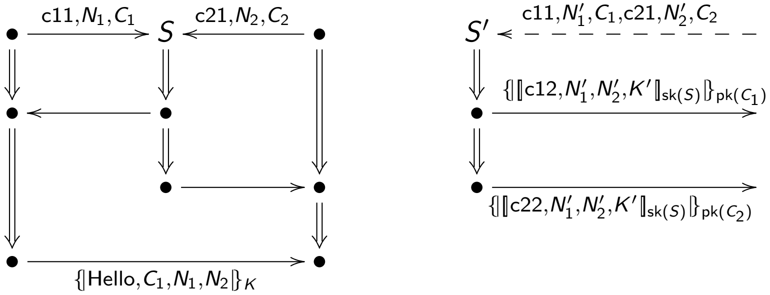
"Bundle" formalizes execution



This interaction
is inessential

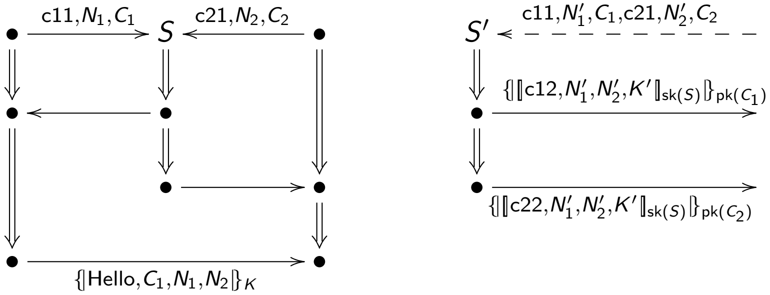
Uncoupling $\mathcal{B}$ into $\mathcal{C}$ via a renaming

$[N_2 \mapsto N_1',\ N_1 \mapsto N_2']$ on right



Uncoupling $\mathcal{B}$ into $\mathcal{C}$ via a renaming

$[N_2 \mapsto N_1',\ N_1 \mapsto N_2']$ on right



Actually, $\mathcal{B}$ results from this $\mathcal{C}$ via $\alpha =$
$[N_1 \mapsto N_1,\ N_2 \mapsto N_2,\ N_1' \mapsto N_2,\ N_2' \mapsto N_1]$

"Lies below"

We say that $\mathcal{C}$ lies below $\mathcal{B}$ via α

1. α is a "local renaming" relative to a partition of $\mathcal{C}$
2. $\mathcal{B}$ has all the edges of $\mathcal{C}$ and possibly more

"Lies below"

We say that $\mathcal{C}$ lies below $\mathcal{B}$ via α

1. α is a "local renaming" relative to a partition of $\mathcal{C}$
2. $\mathcal{B}$ has all the edges of $\mathcal{C}$ and possibly more

Special kind of homomorphism $\mathcal{C} \xrightarrow{\alpha} \mathcal{B}$

"Lies below"

We say that $\mathcal{C}$ lies below $\mathcal{B}$ via α

1. α is a "local renaming" relative to a partition of $\mathcal{C}$
2. $\mathcal{B}$ has all the edges of $\mathcal{C}$ and possibly more

Special kind of homomorphism $\mathcal{C} \stackrel{\alpha}{\rightarrow} \mathcal{B}$

$\mathcal{C}$ is lower in the "information ordering:"

- Fewer arrows
- Fewer identifications of parameters

Session Behavior, v. 1

Relates local runs of participants to global session

A protocol Π has session behavior if, in every execution:

- Any two complete local runs that interact belong to the same global session — homogeneity
- Any two complete local runs of the same role belong to different global sessions — exclusion

Even with active, malicious adversary

Whatever the adversary can do,
he can do without confusing sessions

May-influence relations

$\rightsquigarrow$ is a "may-influence" relation if

preorder: $\rightsquigarrow$ is a preorder on nodes

progress: $m \Rightarrow n$ implies $m \rightsquigarrow n$

no Vs: When a uniquely originates at n_1 and reaches n_2:

$$\begin{array}{ll} n_1 \rightsquigarrow n_3 \\ n_2 \rightsquigarrow n_3 \end{array} \quad \text{implies} \quad \begin{array}{ll} n_1 \rightsquigarrow n_3 \\ n_1 \rightsquigarrow n_2 \\ n_2 \rightsquigarrow n_3 \end{array}$$

Session Behavior, v. 2

- Π obeys $\rightsquigarrow$ if,
  for all bundles $\mathcal{B}$, there exists a $\mathcal{C}$ lying below $\mathcal{B}$ s.t.

$$m \preceq_{\mathcal{C}} n \quad \text{implies} \quad m \rightsquigarrow n.$$

- The session influence relation, $m \rightsquigarrow_{\mathsf{s}} n$, holds, if

  every session parameter a_ρ
  already defined on m is defined on n,
  and takes the same value on n

- Π has session behavior if

$$\Pi \text{ obeys } \rightsquigarrow_{\mathsf{s}}$$

So why is this hard?

- Syntactic flexibility
  - Session parameters may contribute to plaintext or key
  - Participants may join late
- Adversary model: Partial compromise
  - Compliant participants get session behavior, despite compromised participants in session

double-commission problem

- "Same session" is not an equivalence relation
  - "May influence" is a preorder on nodes
  - This is also an opportunity for generality: Reason about all may-influence relations

Reasoning about separability

- Adversary transports information along paths

Reasoning about separability

- Adversary transports information along paths
- Progressively takes apart
  and reconstructs messages

Reasoning about separability

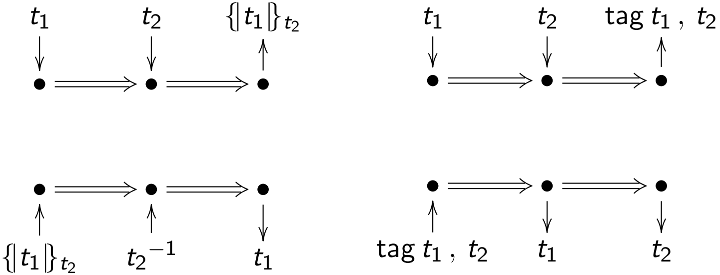
- Adversary transports information along paths
- Progressively takes apart
  and reconstructs messages
- If adversary breaks down to basic parts,
  can substitute new basic values

Reasoning about separability

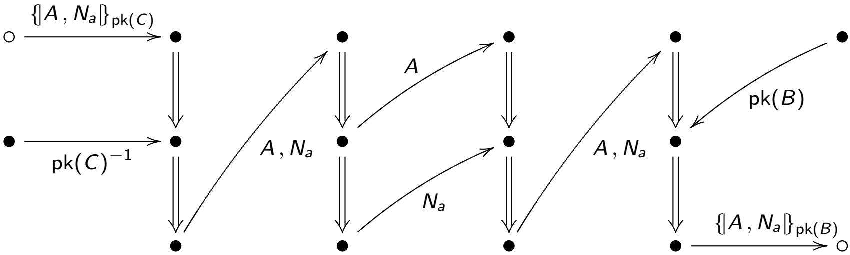
- Adversary transports information along paths
- Progressively takes apart
  and reconstructs messages
- If adversary breaks down to basic parts,
  can substitute new basic values
- Assuming that results he delivers
  never get back to original source

Adversary actions

$$\bullet \xrightarrow{a}$$



Adversary paths



About Adversary Paths

- Path p is direct[1] if it has no key edges, except maybe the last step

[1] This talk: only consider direct paths; applicable when protocols use basic keys.

About Adversary Paths

- Path p is direct[1] if it has no key edges, except maybe the last step
- Path p is normal if any destruction precedes all construction

[1] This talk: only consider direct paths; applicable when protocols use basic keys.

About Adversary Paths

- Path p is direct[1] if it has no key edges, except maybe the last step
- Path p is normal if any destruction precedes all construction
- Bundle $\mathcal{B}$ is normal if all its direct paths are normal

[1] This talk: only consider direct paths; applicable when protocols use basic keys.

About Adversary Paths

- Path p is direct[1] if it has no key edges, except maybe the last step
- Path p is normal if any destruction precedes all construction
- Bundle $\mathcal{B}$ is normal if all its direct paths are normal
- The bridge of a normal path is the message that follows all destruction and precedes all construction

[1] This talk: only consider direct paths; applicable when protocols use basic keys.

A lemma

Lemma

1. *Every bundle is equivalent to a normal bundle*

A lemma

Lemma

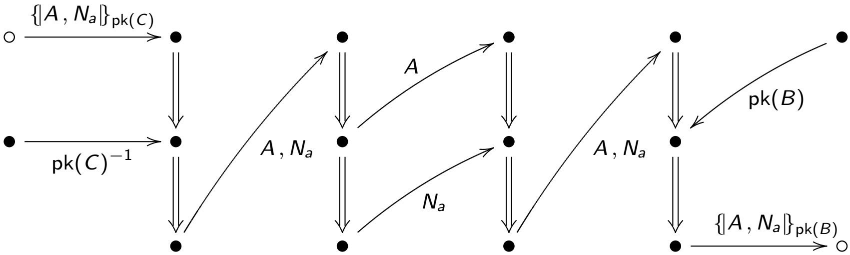
1. *Every bundle is equivalent to a normal bundle*
2. *Let p be direct and normal.*
    1. *The bridge of p is a common ingredient of the messages at the two ends*

A lemma

Lemma

1. *Every bundle is equivalent to a normal bundle*
2. *Let p be direct and normal.*
   1. *The bridge of p is a common ingredient of the messages at the two ends*
   2. *Either some encryption $\{|t|\}_K$ appears at both ends of p, or else the bridge of p is encryption-free*

Adversary paths



A lemma

Lemma

1. *Every bundle is equivalent to a normal bundle*
2. *Let p be direct and normal.*
   1. *The bridge of p is a common ingredient of the messages at the two ends*
   2. *Either some encryption $\{\!|t|\!\}_K$ appears at both ends of p, or else the bridge of p is encryption-free*

A lemma

Lemma

1. *Every bundle is equivalent to a normal bundle*
2. *Let p be direct and normal.*
   1. *The bridge of p is a common ingredient of the messages at the two ends*
   2. *Either some encryption* $\{\!|t|\!\}_K$ *appears at both ends of p, or else the bridge of p is basic*

Paths that cross sessions

Suppose in $\mathcal{B}$

1. All session parameters appear in every encryption $\{|t|\}_K$
2. p crosses from one session to another, i.e.

$$\mathsf{first}(p) \not\rightsquigarrow_{\mathsf{s}} \mathsf{last}(p)$$

Then

1. p has basic bridge term a

Paths that cross sessions

Suppose in $\mathcal{B}$

1. All session parameters appear in every encryption $\{\!|t|\!\}_K$
2. p crosses from one session to another, i.e.

$$\mathsf{first}(p) \not\rightarrow_{\mathsf{s}} \mathsf{last}(p)$$

Then

1. p has basic bridge term a
2. Adversary can omit edge and replace a with some other a'

Paths that cross sessions

Suppose in $\mathcal{B}$

1. All session parameters appear in every encryption $\{\!|t|\!\}_K$
2. p crosses from one session to another, i.e.

$$\mathsf{first}(p) \not\rightarrowtail_{\mathsf{s}} \mathsf{last}(p)$$

Then

1. p has basic bridge term a
2. Adversary can omit edge and replace a with some other a'
3. Resulting bundle $\mathcal{C}$ lies below $\mathcal{B}$

Paths that cross sessions

Suppose in $\mathcal{B}$

1. All session parameters appear in every encryption $\{|t|\}_K$
2. p crosses from one session to another, i.e.

$$\text{first}(p) \not\rightsquigarrow_{\mathsf{s}} \text{last}(p)$$

Then

1. p has basic bridge term a
2. Adversary can omit edge and replace a with some other a'
3. Resulting bundle $\mathcal{C}$ lies below $\mathcal{B}$
4. No-Vs property implies change never affects later node

The Separability Theorem

Definition

1. Path p is $\rightsquigarrow$-critical if first(p) $\not\rightsquigarrow$ last(p)
2. $\mathcal{B}$ is $\rightsquigarrow$-reparable if every $\rightsquigarrow$-critical path in $\mathcal{B}$ has a basic bridge

The Separability Theorem

Definition

1. Path p is $\rightsquigarrow$-critical if first$(p) \not\rightsquigarrow$ last(p)
2. $\mathcal{B}$ is $\rightsquigarrow$-reparable if every $\rightsquigarrow$-critical path in $\mathcal{B}$ has a basic bridge

Theorem

For every $\rightsquigarrow$-reparable $\mathcal{B}$,
there is a $\mathcal{C}$ lying below $\mathcal{B}$ such that $\mathcal{C}$ obeys $\rightsquigarrow$

The Separability Theorem

Definition

1. Path p is $\rightsquigarrow$-critical if $\text{first}(p) \not\rightsquigarrow \text{last}(p)$
2. $\mathcal{B}$ is $\rightsquigarrow$-reparable if every $\rightsquigarrow$-critical path in $\mathcal{B}$ has a basic bridge

Theorem

For every $\rightsquigarrow$-reparable $\mathcal{B}$,
there is a $\mathcal{C}$ lying below $\mathcal{B}$ such that $\mathcal{C}$ obeys $\rightsquigarrow$

To prove separability, check:

1. $m \not\rightsquigarrow n$ implies no common encryptions
2. $\rightsquigarrow$ satisfies no-Vs

may involve restrictions

Applications: Sessions

Session protocols: Π-bundles are $\rightsquigarrow_s$-reparable if Π has session parameters

No-Vs property requires a restriction,
that some keys uncompromised,
when Π allows late arrivals

Applications, 2

Protocol composition, I: $\Pi_1 \cup \Pi_2$-bundles are $\rightsquigarrow_C$-reparable if
Π_1 and Π_2 have no unifiable encryptions, and
use only basic keys

$m \rightsquigarrow_C n$ iff $m, n \in \text{nodes}(\Pi_1)$ or $m, n \in \text{nodes}(\Pi_2)$
$\Pi_1 \not\rightsquigarrow_C \Pi_2$, $\Pi_2 \not\rightsquigarrow_C \Pi_1$

Applications, 2

Protocol composition, I: $\Pi_1 \cup \Pi_2$-bundles are $\rightsquigarrow_C$-reparable if
Π_1 and Π_2 have no unifiable encryptions, and
use only basic keys

$m \rightsquigarrow_C n$ iff $m, n \in \text{nodes}(\Pi_1)$ or $m, n \in \text{nodes}(\Pi_2)$
$\Pi_1 \not\rightsquigarrow_C \Pi_2$, $\Pi_2 \not\rightsquigarrow_C \Pi_1$

Protocol composition, II: *No unifiable encryptions* suffices in I

Applications, 2

Protocol composition, I: $\Pi_1 \cup \Pi_2$-bundles are $\rightsquigarrow_C$-reparable if
Π_1 and Π_2 have no unifiable encryptions, and
use only basic keys

$m \rightsquigarrow_C n$ iff $m, n \in \text{nodes}(\Pi_1)$ or $m, n \in \text{nodes}(\Pi_2)$
$\Pi_1 \not\rightsquigarrow_C \Pi_2$, $\Pi_2 \not\rightsquigarrow_C \Pi_1$

Protocol composition, II: *No unifiable encryptions* suffices in I

Protocol composition, III: $\Pi_1 \cup \Pi_2$-bundles are $\rightsquigarrow_{DE}$-reparable if
Π_2 generates no encryptions accepted on Π_1 nodes
Π_2 extracts nothing from Π_1 encryptions

$m \rightsquigarrow_{DE} n$ iff $m \in \text{nodes}(\Pi_1)$ or $m, n \in \text{nodes}(\Pi_2)$
$\Pi_2 \not\rightsquigarrow_C \Pi_1$ but $\Pi_1 \rightsquigarrow_C \Pi_2$

Goals of this hour

- Define session behavior
- Give syntactic criteria for a protocol to have session behavior
- Develop proof methods "Separability theorem"
- Free bonus:

Protocol independence results also follow from separability theorem