Introduction to Computational Soundness (II)

Bogdan Warinschi

- University of Bristol -

The Needham-Schroeder public key protocol

A

B

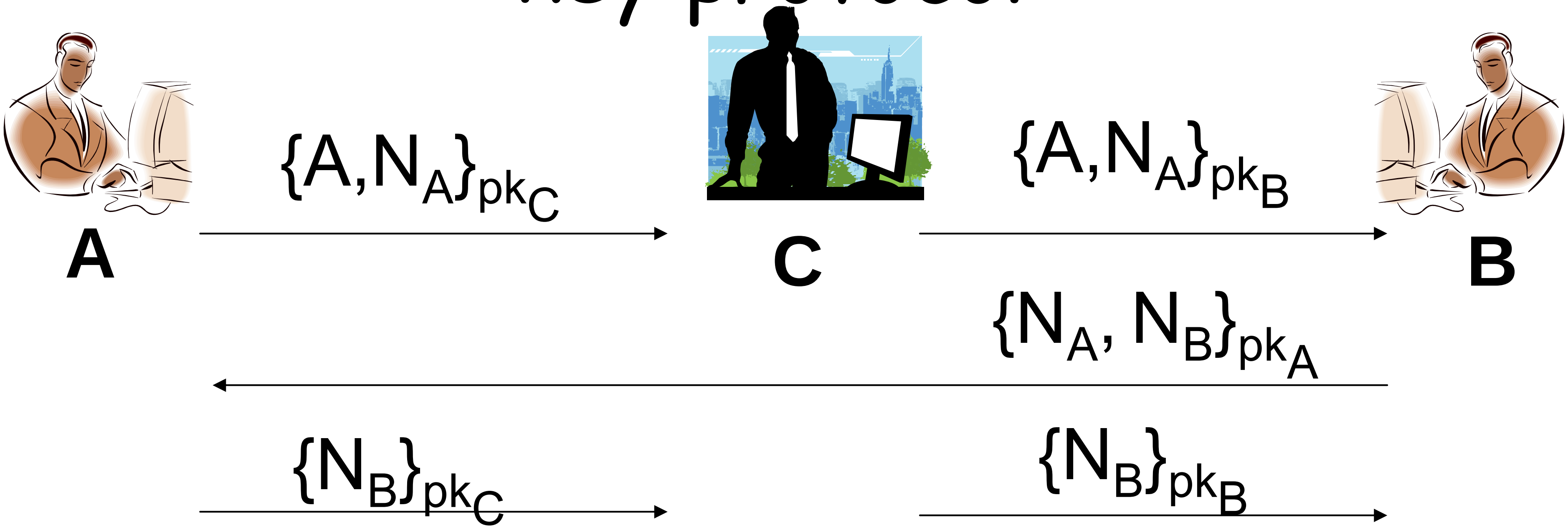
$$A \xrightarrow{\{A, N_A\}_{pk_B}} B$$

$$A \xleftarrow{\{N_A, N_B\}_{pk_A}} B$$

$$A \xrightarrow{\{N_B\}_{pk_B}} B$$

- **Nonce N_B sent in the second message:**
  - **is intended for *A* (identity received in the first message)**
  - **should be secret to any other party but *A***
- ***A* and B should have matching conversations**

The Needham-Schroeder public key protocol



- N_B is secret if the adversary is passive
- N_B is not secret if the adversary is active
- Matching conversations does not hold

Lowe's fix - Secure Version of NS

$$\xrightarrow{\{A,N_A\}_{pk_B}}$$

$$\xleftarrow{\{B,N_A,N_B\}_{pk_A}}$$

$$\xrightarrow{\{N_B\}_{pk_B}}$$

No more "logical" attacks; protocol secure

… or is it?

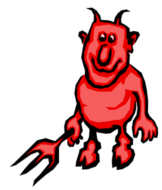
Implement the protocol with an (IND-CPA) secure encryption scheme "

Adv (" ,A)(¿)=

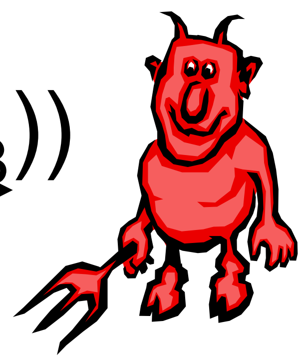
Pr[(pk,sk) ″ K(¿): A$^{\text{E(pk,.)}}$ =1] –

Pr[(pk,sk) ″ K(¿): A$^{\text{E(pk,0}^{|ƒ|}\text{)}}$=1]

Another gap

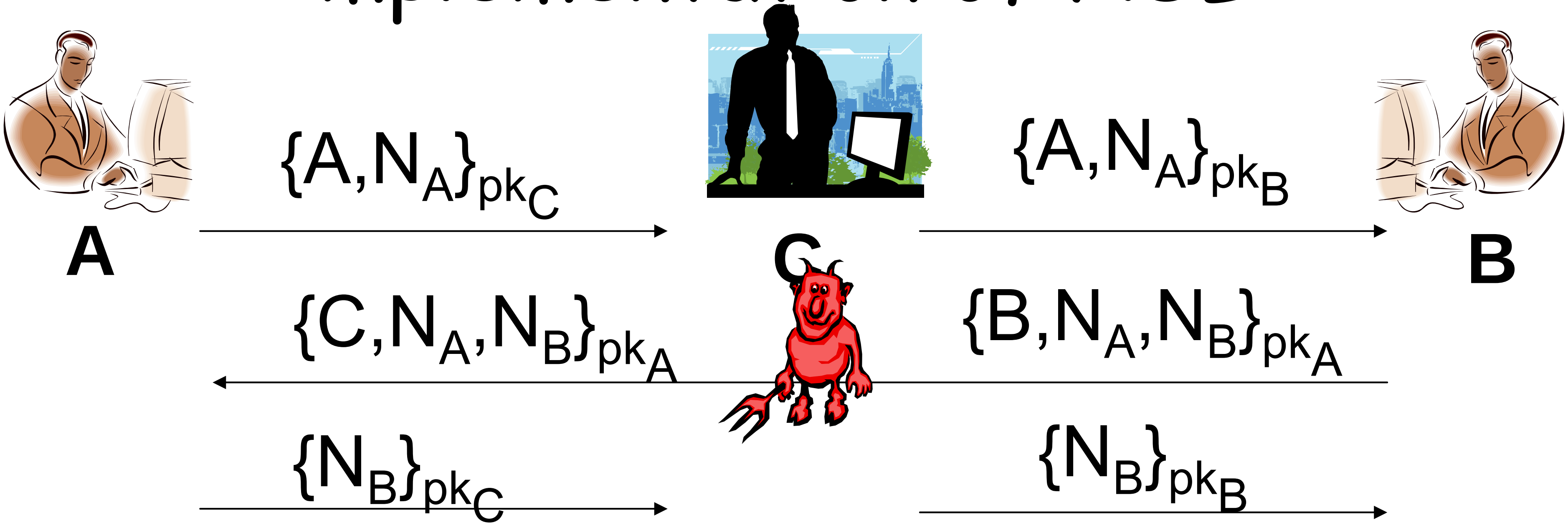
- There exist IND-CPA secure encryption scheme and a deterministic polynomial time algorithm  such that

$E(pk_A, (B, N_A\, N_B))$



$E(pk_A, (C, N_A\, N_B))$

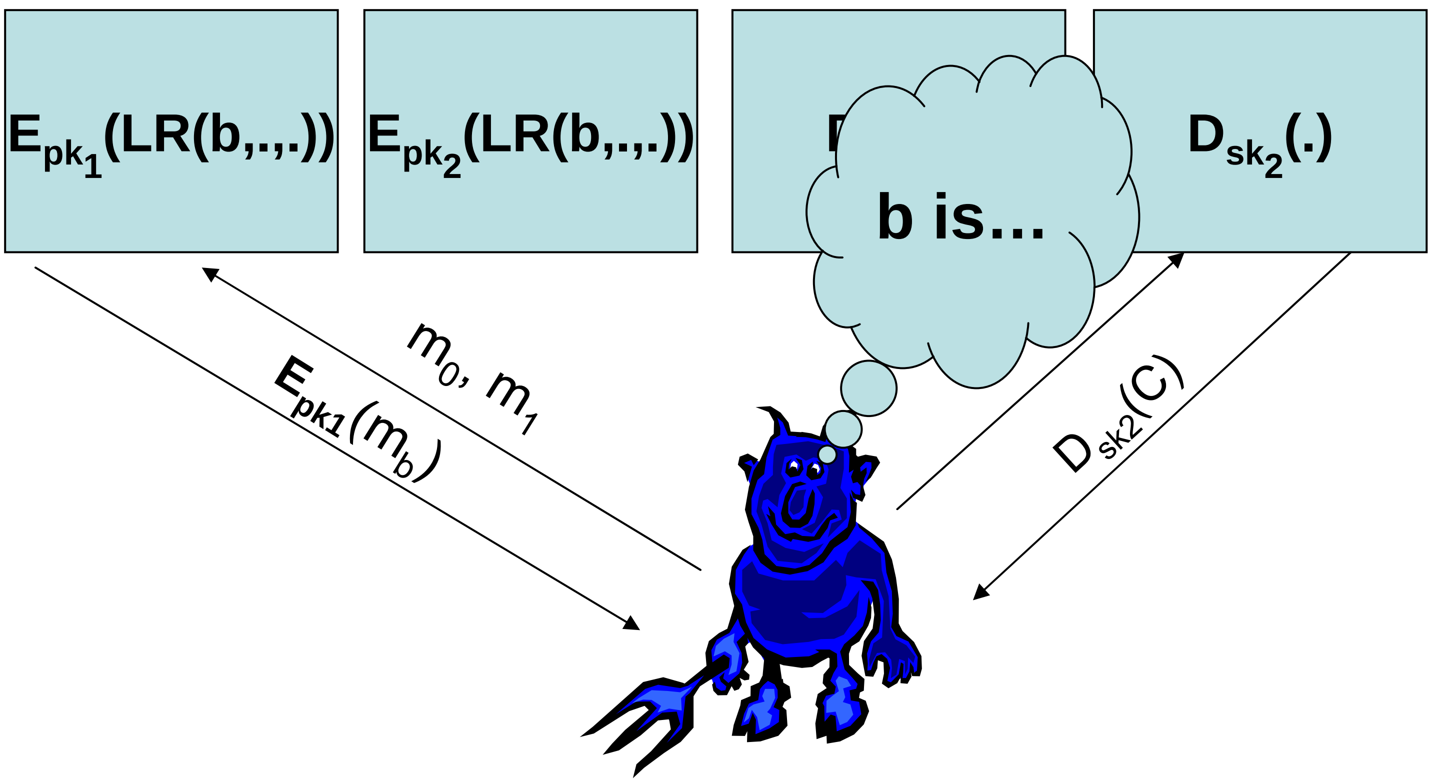
An attack against an implementation of NSL



- N_B may not be secret even if encryption is IND-CPA
- Matching conversations does not hold
- … use stronger encryption

IND-CCA security for multi-users

- Implement encryption with a scheme (K,E,D) that is IND-CCA secure



…back to NSL

- If NSL is implemented with an encryption scheme that is IND-CCA secure then:
  - N_B is secret
  - Matching conversations holds

A gap

- Security of primitives is
  - *axiomatized* (in the symbolic approach)
  - *defined* (in the computational approach)
- Question:
  - Symbolically: not possible to calculate $\{C,N_A,N_B\}_{pk_A}$ out of $\{B,N_A,N_B\}_{pk_A}$
  - Computationally: is it possible to enforce the above?

Computational soundness

- The goal is to find sufficient security conditions on the primitives used in the implementation such that a protocol secure in the symbolic setting is also secure in the computational setting…
- …but what is a protocol, what does secure mean?

Protocols

- A sequence of message exchanges
- Messages constructed from constants, variables, and cryptographic operations

Send $\{A,N_A\}_{pk_B}$

Receive $\{B,N_A,X\}_{pk_A}$

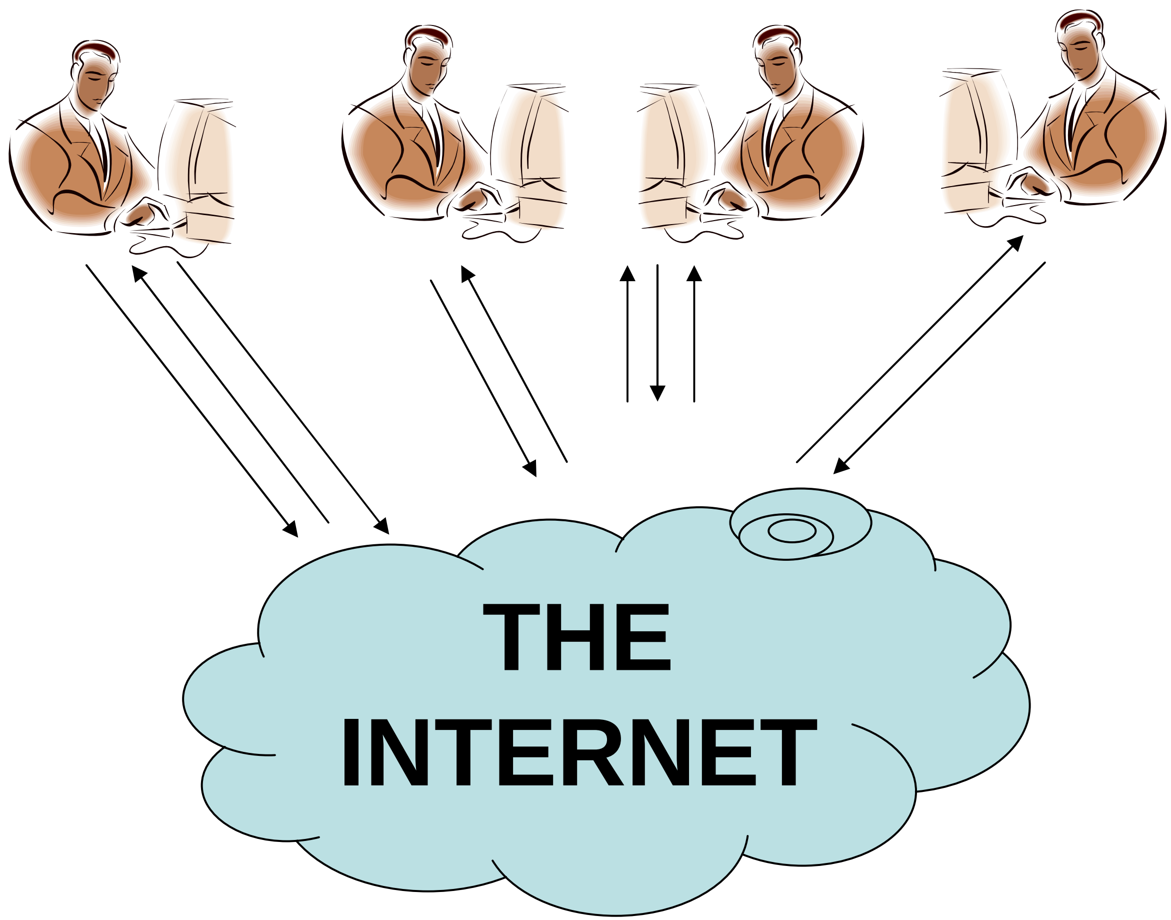
Send $\{X\}_{pk_B}$

Receive $\{A,Y\}_{pk_B}$

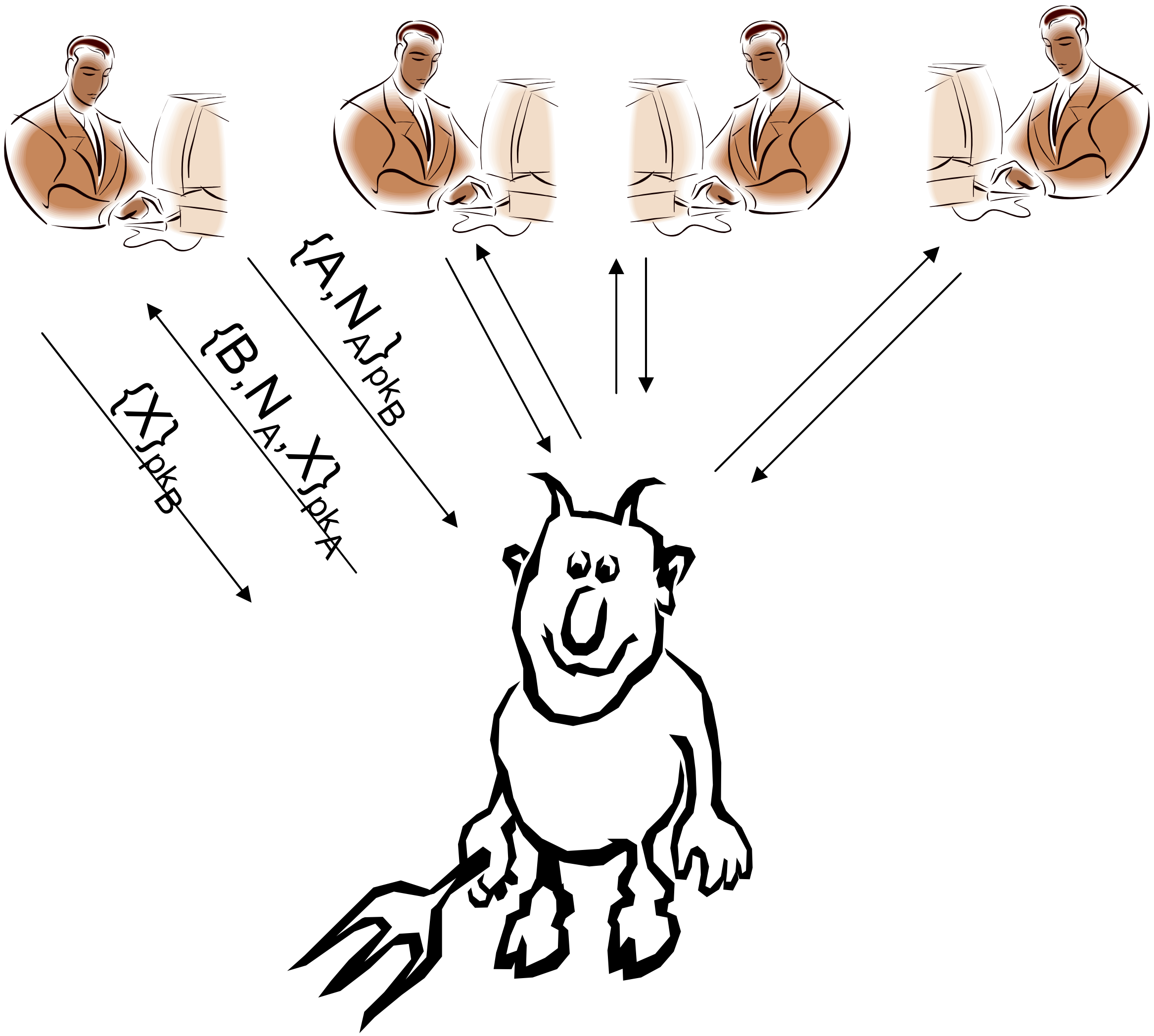
Send $\{B,Y,N_B\}_{pk_A}$

Receive $\{N_B\}_{pk_B}$

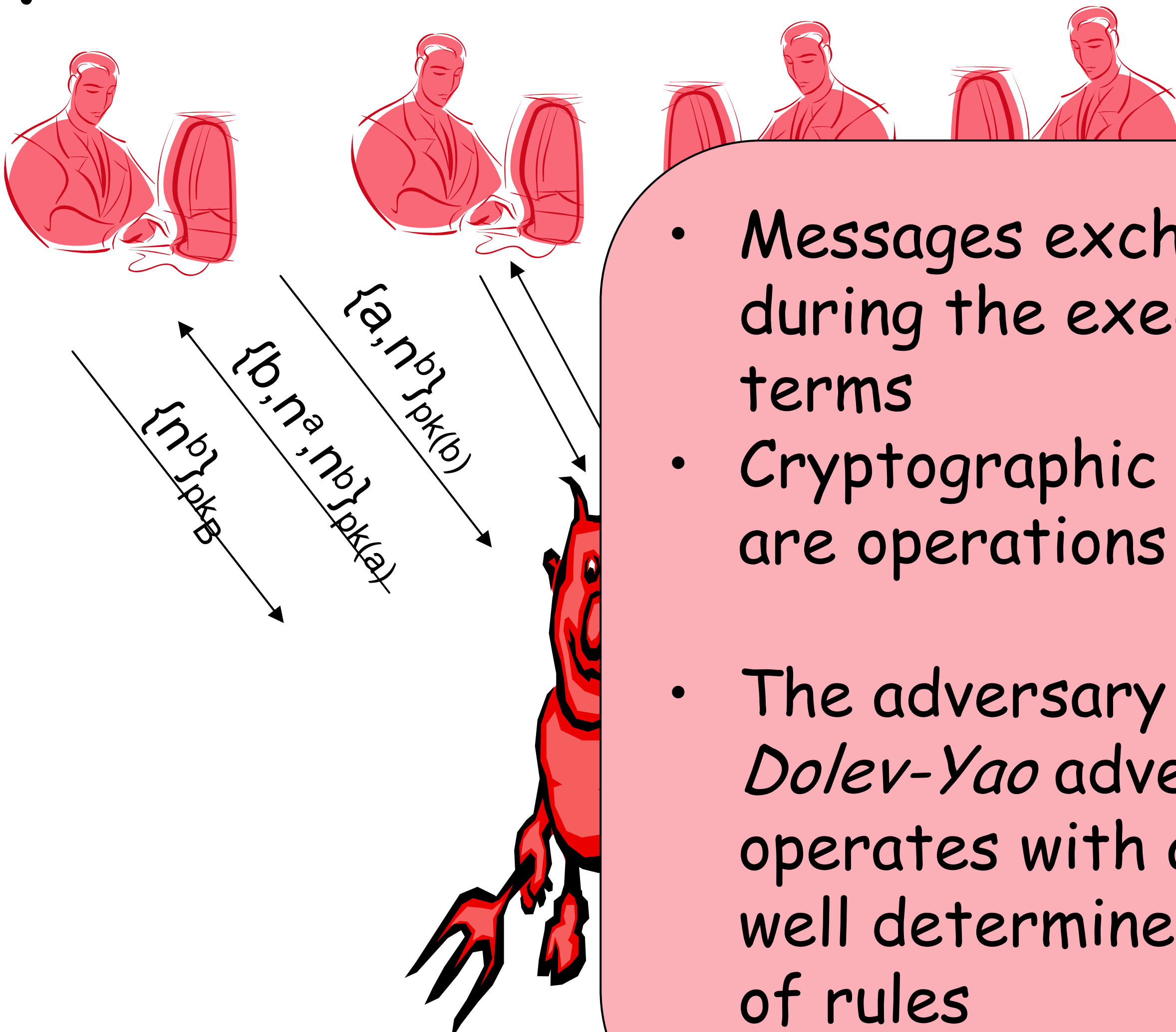
Communication is over a network



(Generic) Execution model



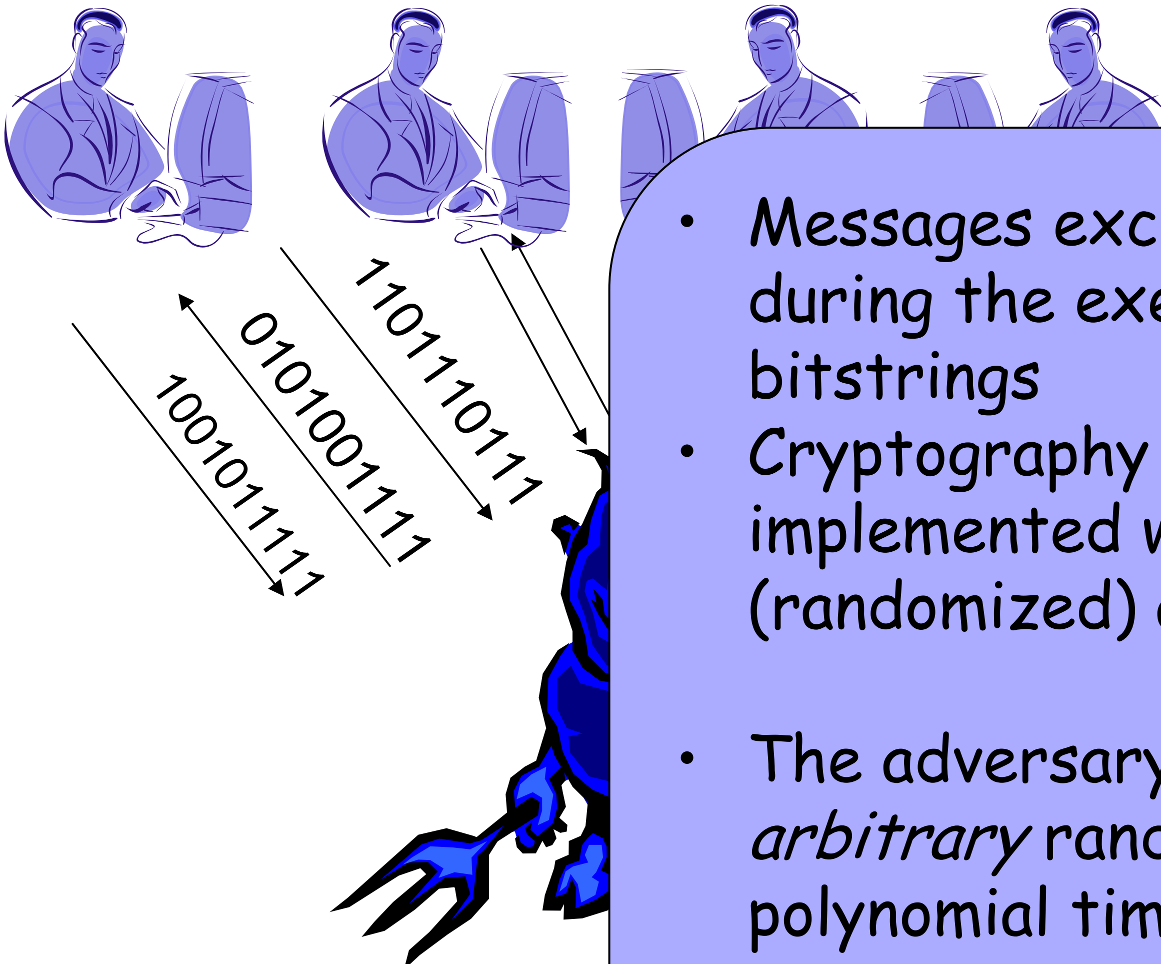
Symbolic execution model



- Messages exchanged during the execution are terms
- Cryptographic operations are operations on terms

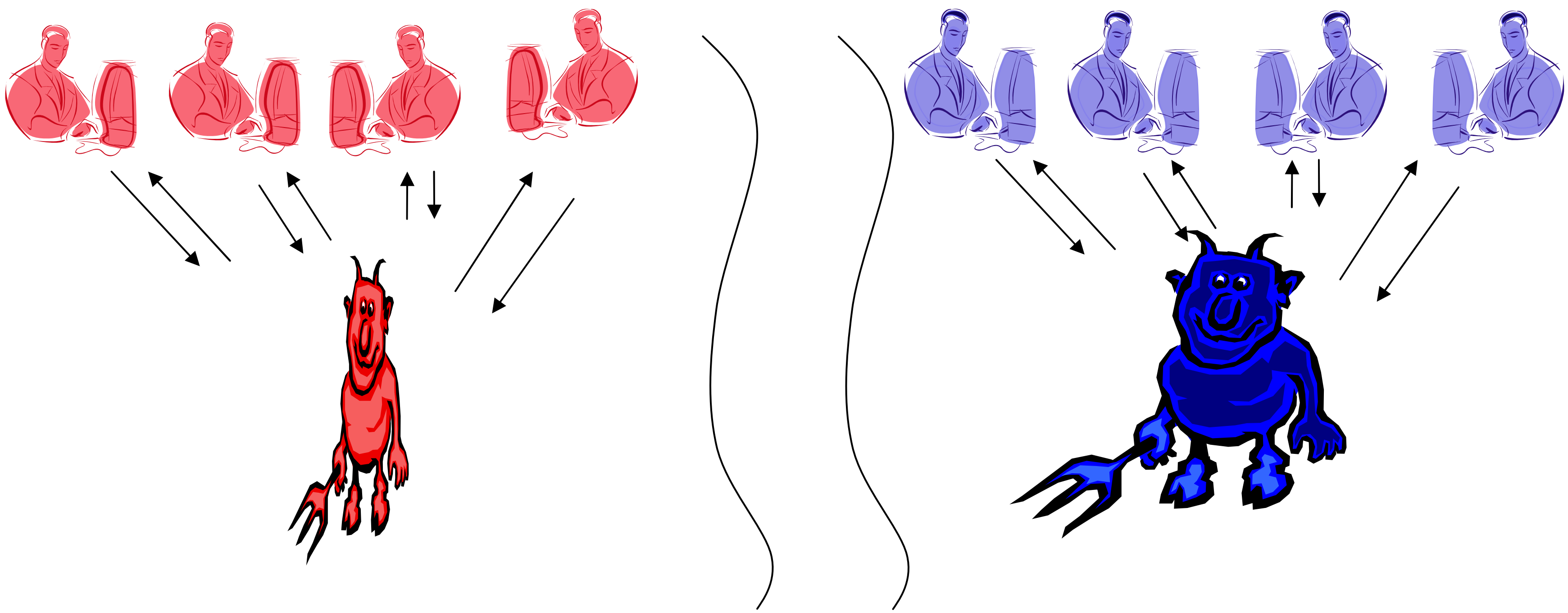
- The adversary is a *Dolev-Yao* adversary who operates with a finite, well determined number of rules

Computational execution model



- Messages exchanged during the execution are bitstrings
- Cryptography implemented with actual (randomized) algorithms
- The adversary is an *arbitrary* randomized polynomial time algorithm

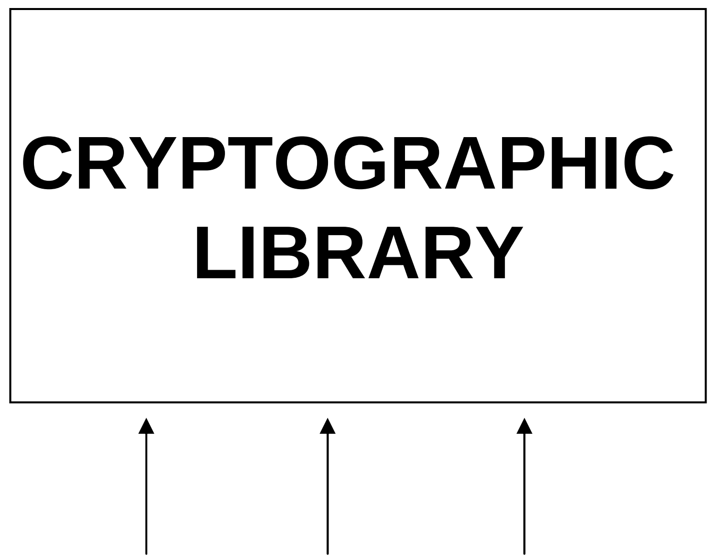
Back to the gap



- Security properties are statements about two very different executions
    - Non-deterministic executions (symbolically)
    - Randomized executions (computationally)

Computational soundness via black-box reactive simulation

The simulation approach
[Backes, Pfitzmann, Waidner]



Nonce generation, Encryption,
Decryption, Signing, MACs, etc

The simulation approach

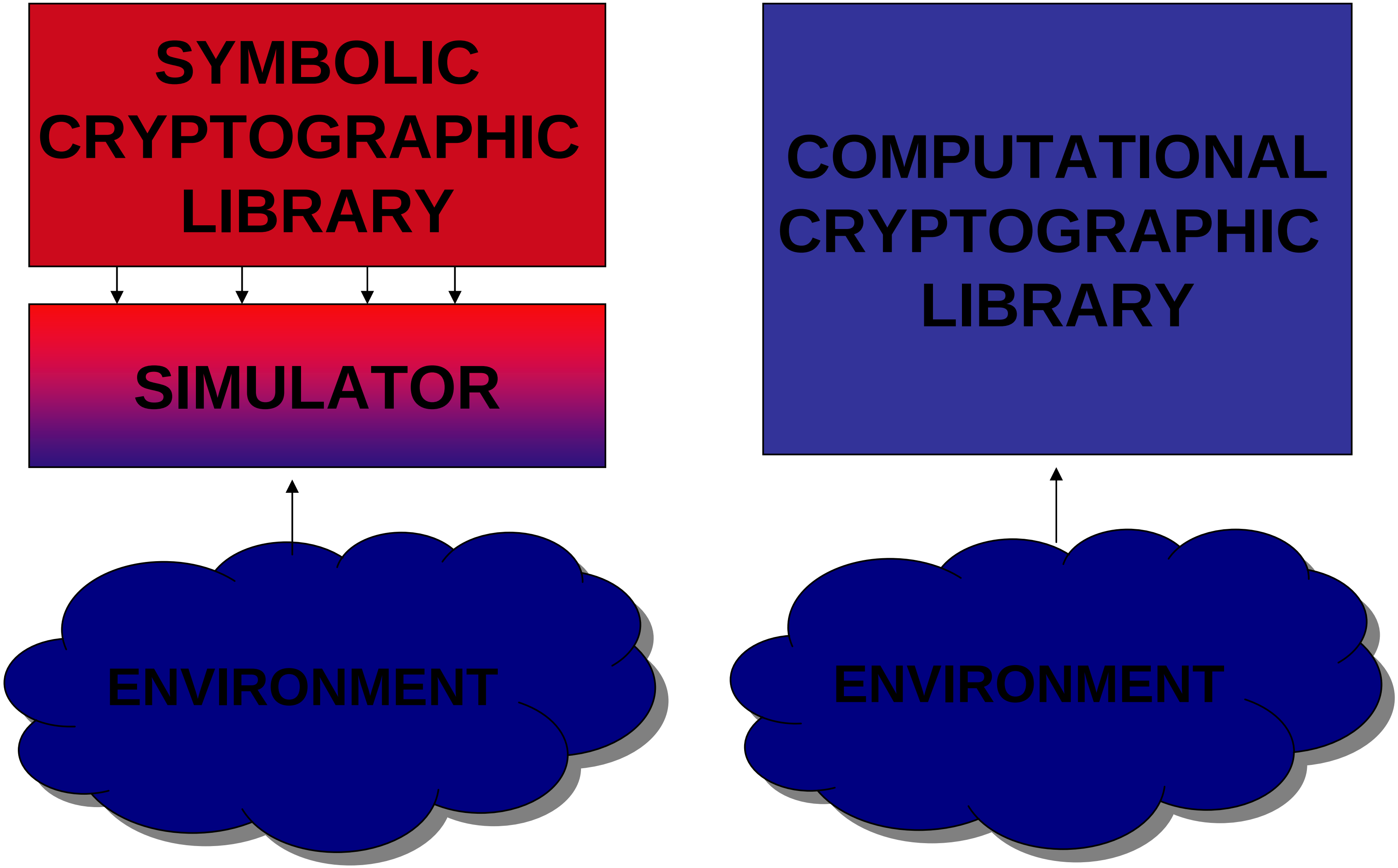
[Backes, Pfitzmann, Waidner]

SYMBOLIC CRYPTOGRAPHIC LIBRARY

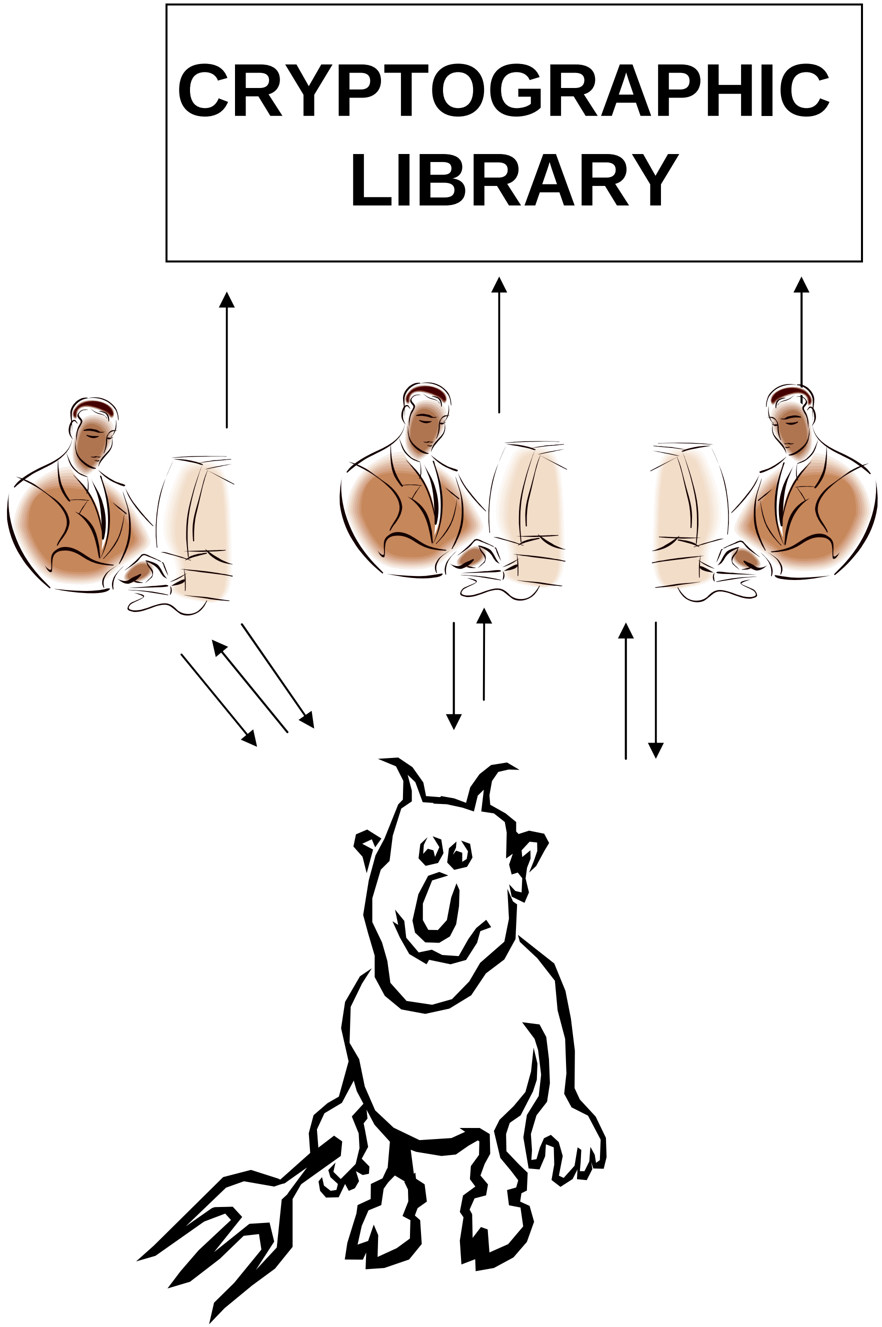
COMPUTATIONAL CRYPTOGRAPHIC LIBRARY

Internally the library operates with terms and enforces Dolev-Yao behaviours

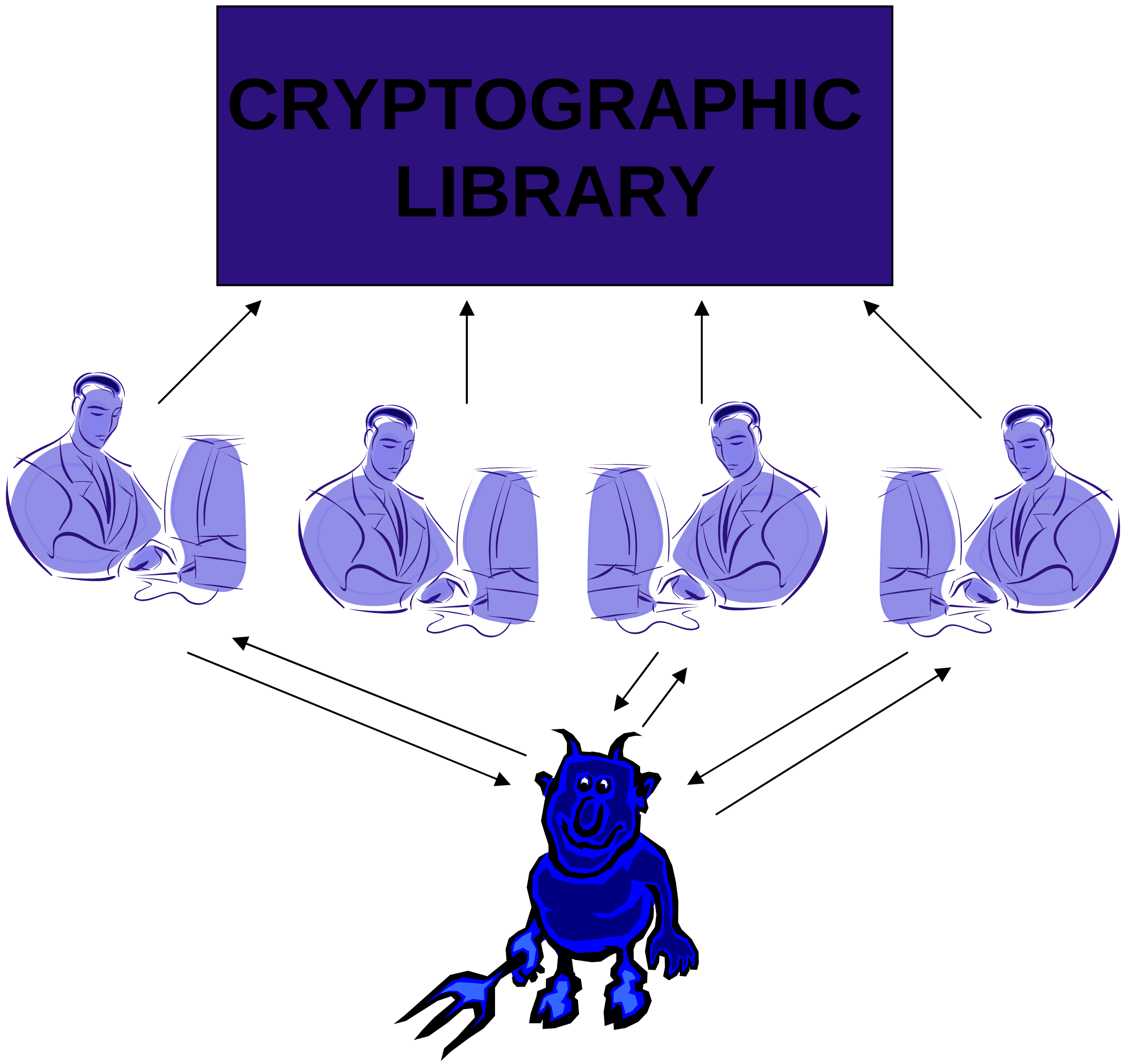
Internally the library operates with bitstrings and actual cryptographic algorithms



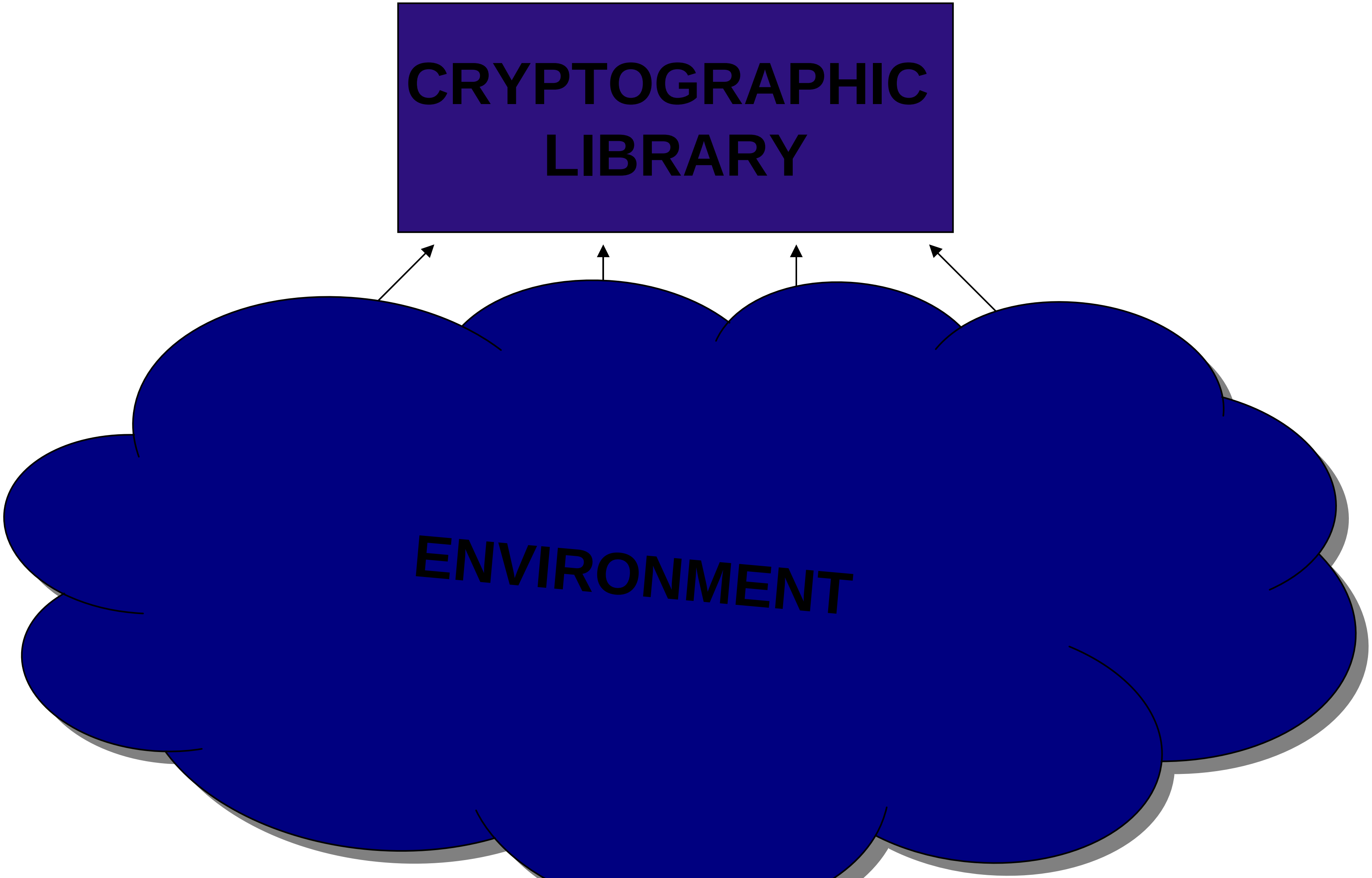
THEOREM: If cryptographic primitives are secure in the computational cryptographic library, then there exists a simulator such that no probabilistic polynomial time environment can distinguish between the two worlds

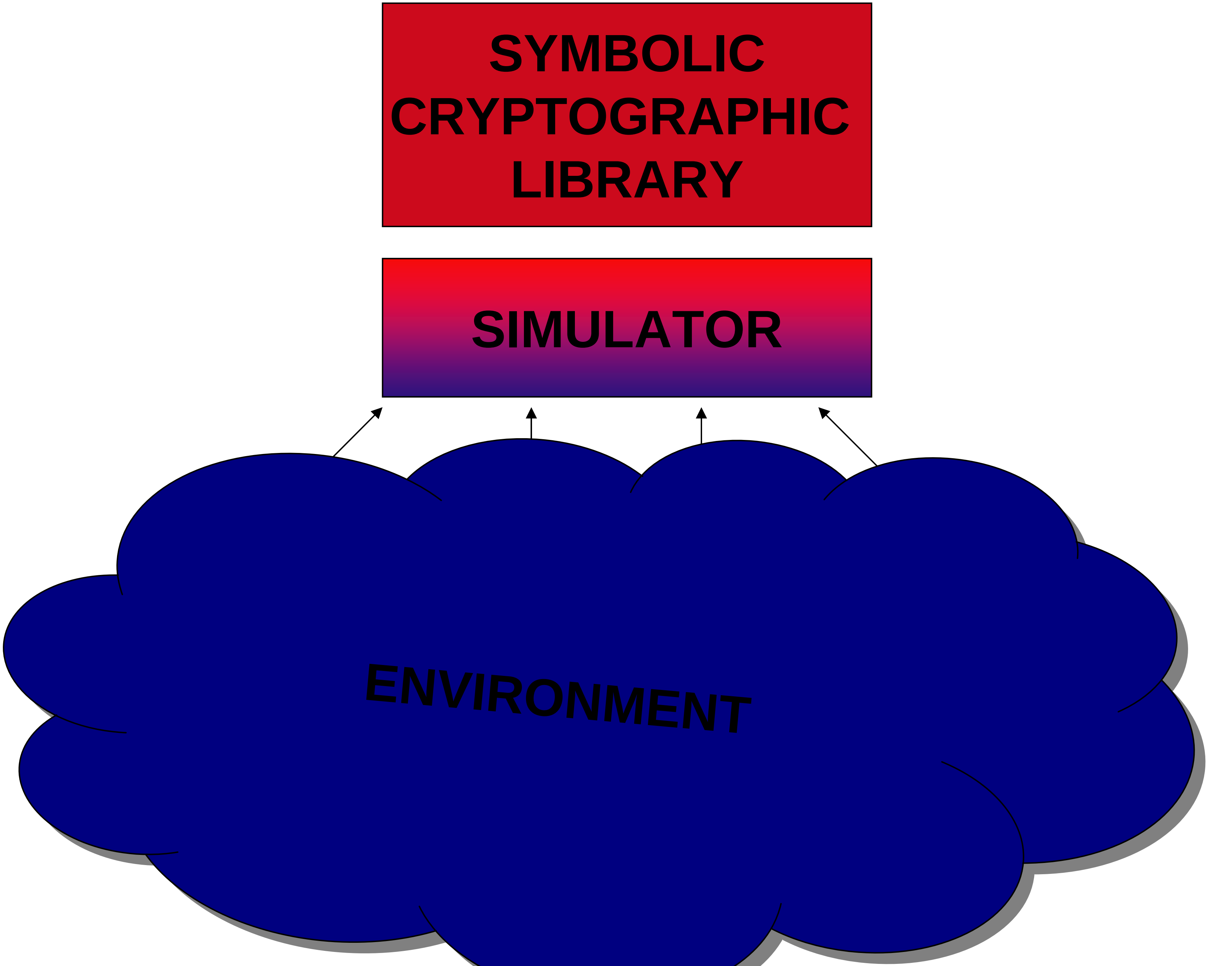

CRYPTOGRAPHIC
LIBRARY

Protocol execution with a cryptographic library



Protocol execution with a cryptographic library




SYMBOLIC CRYPTOGRAPHIC LIBRARY
SIMULATOR
ENVIRONMENT

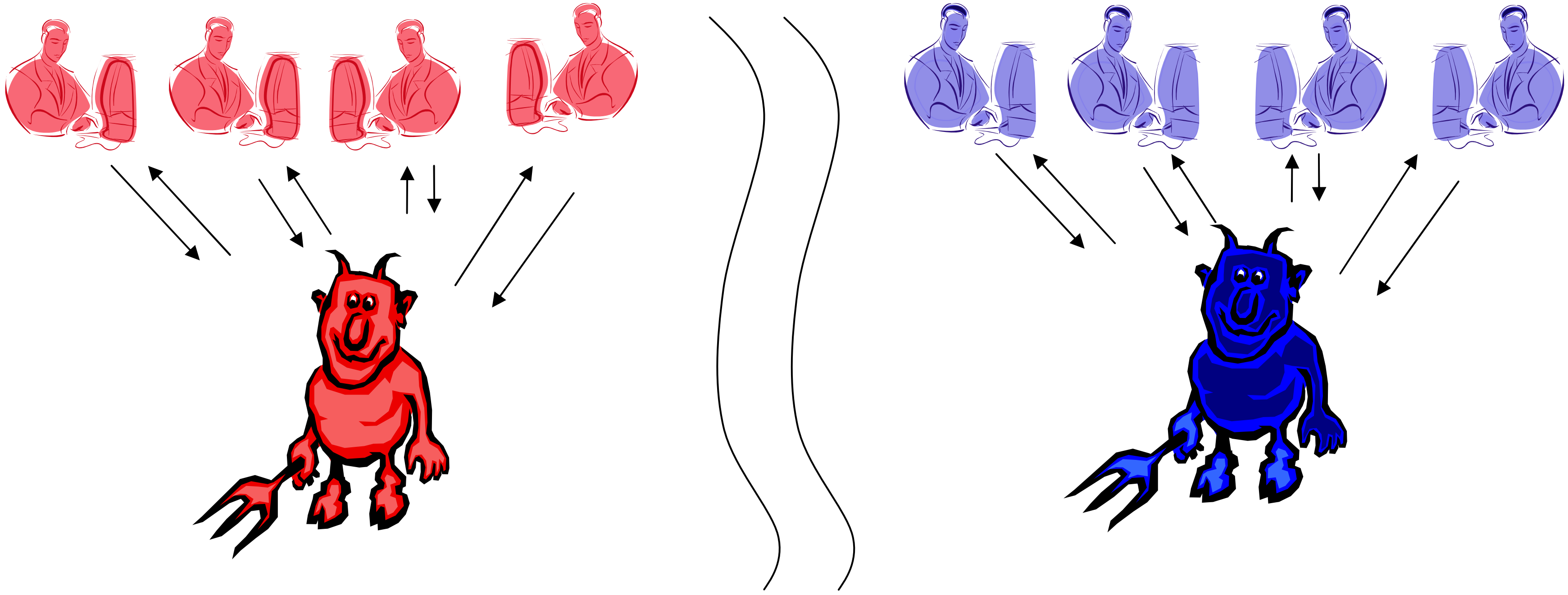
Soundness with a cryptographic library

- Security of protocols can be analyzed in a world where cryptography is idealized in the Dolev-Yao style

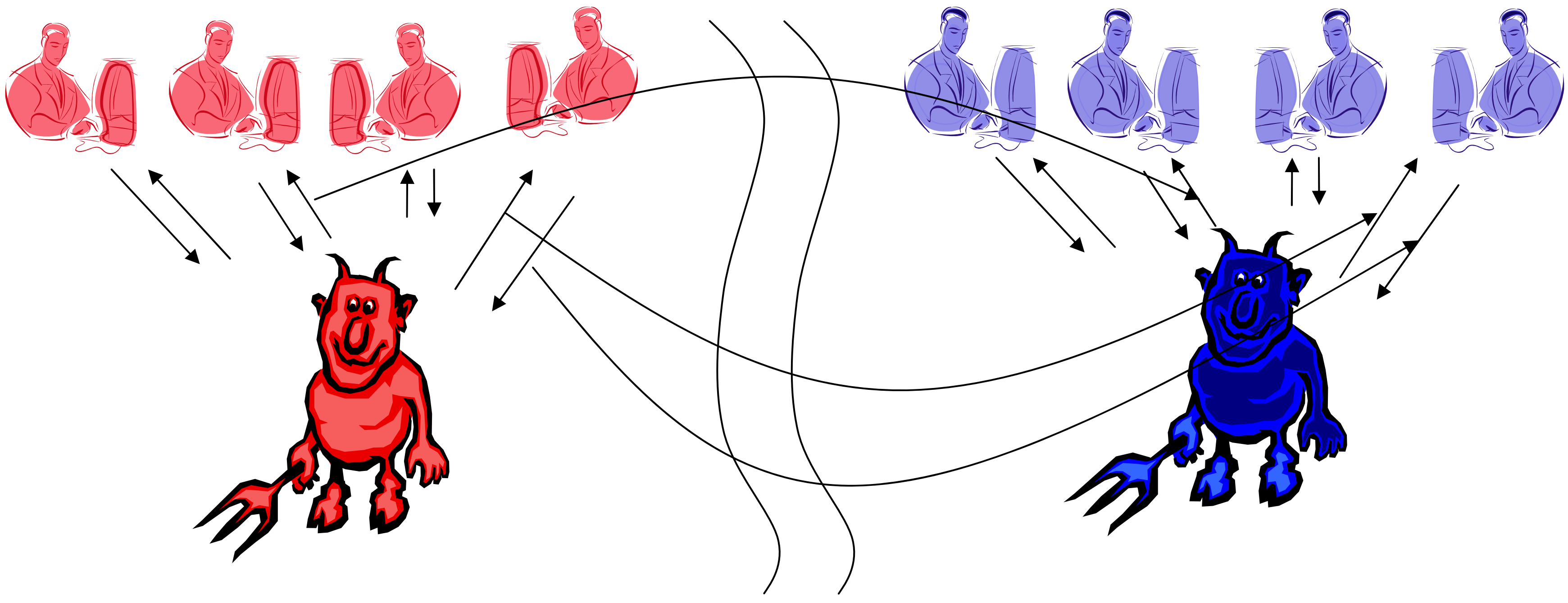
Computational soundness via trace mapping

Trace mapping

[Micciancio, Warinschi]



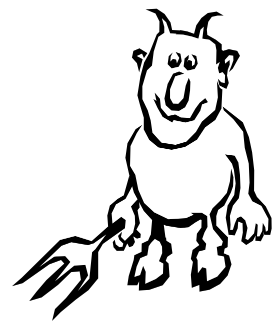
The trace mapping approach



Symbolic execution of a protocol

Real execution of a protocol

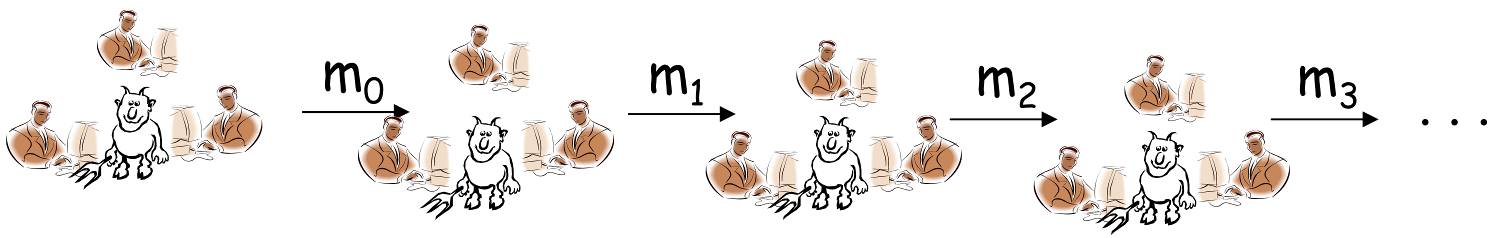
A bit more precisely



- The adversary may be able to corrupt parties
- The adversary may send any message it wants to a session and receives the answer calculated by the session

Execution traces

Execution trace:



Formally

$$F_0 \xrightarrow{m_0} F_1 \xrightarrow{m_1} F_2 \xrightarrow{m_2} F_3 \xrightarrow{m_3} \cdots$$

F_i : Local variables of sessions -> Values

Symbolic executions

$$F_0 \xrightarrow{m_0} F_1 \xrightarrow{m_1} F_2 \xrightarrow{m_2} F_3 \xrightarrow{m_3}$$

- Messages, values etc… are terms

  F_i : Local variables of sessions -> Terms
- Adversary can only send messages that he can compute according to the Dolev Yao rules
- Nondeterministic executions
- For protocol " and adversary *A*, write Tr_s(" ,*A*) for the trace determined by *A*

Computational executions

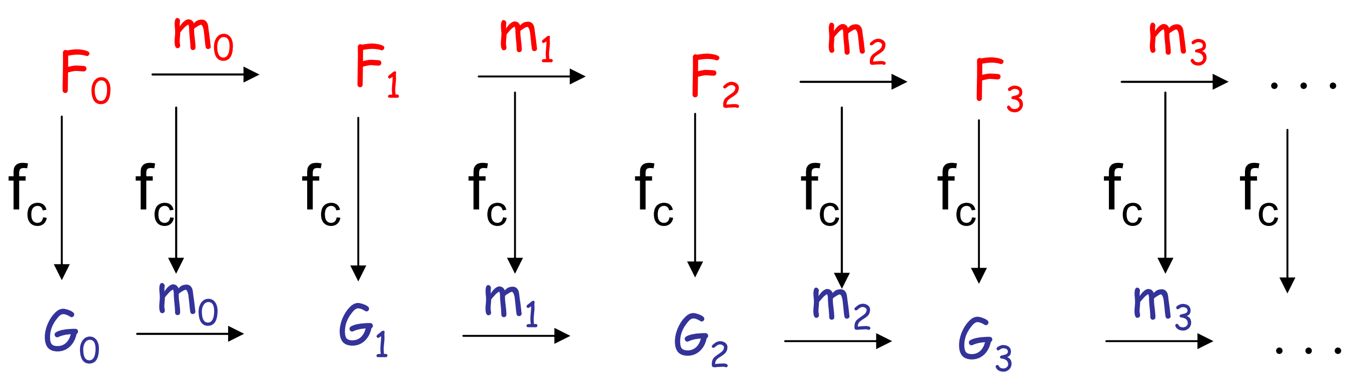
$$G_0 \xrightarrow{m_0} G_1 \xrightarrow{m_1} G_2 \xrightarrow{m_2} G_3 \xrightarrow{m_3} \ldots\ldots$$

- Messages, values etc… are bitstrings

  G_i : Local variables of sessions -> Bitstrings

- Adversary can only send any polynomial-time computable message
- Executions are randomized
- Tr_c(" (R"),$A(R_A)$) is the execution trace determined by adversary A, randomness R" and R_A

Computational soundness result



- **"Mapping lemma"**: With overwhelming probability the computational trace is the image of a Dolev-Yao trace through an appropriate mapping f_c.
- **Interpretation:** The real adversary only performs Dolev Yao operations!!!

Trace mapping lemma

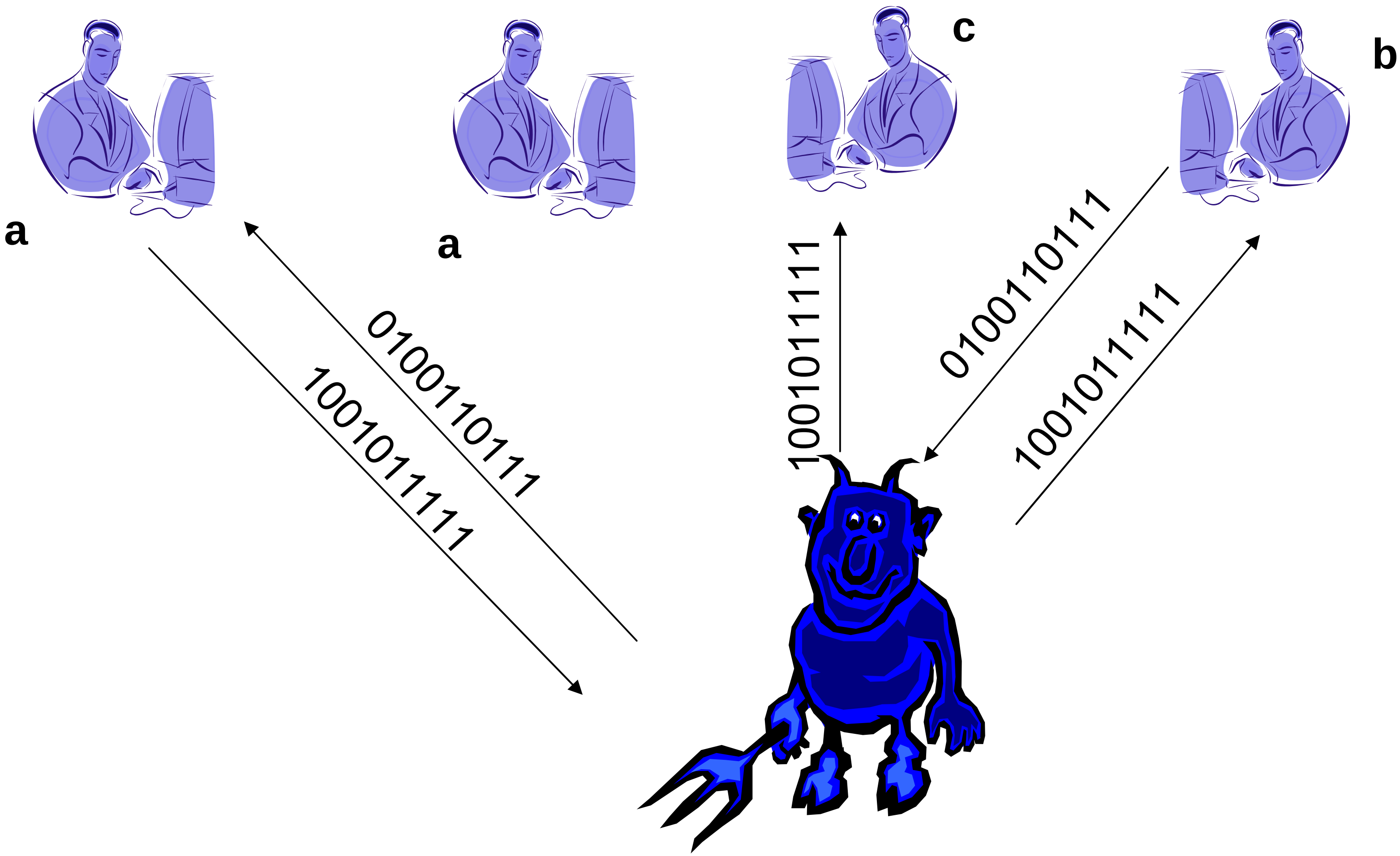
- Let " be a protocol and *A* a computational adversary. If " is implemented with secure primitives then almost all of the computational traces of " are images of symbolic Dolev-Yao traces.

Prob[< B, < f_c : Tr_c(" ($R_{"}$), *A*(R_A)) = f_c(Tr_s(" ,B))]
is overwhelming

Proof idea

1. Fix an adversary A
2. Any concrete execution can be mapped to a symbolic execution
3. Show that this symbolic execution is that of a Dolev-Yao adversary (with overwhelming probability)
   … or otherwise one can use A to break the underlying primitives

Step 2: From concrete executions…



Send $\{A,N_A\}_{pk_B}$

Receive $\{B,N_A,X\}_{pk_A}$

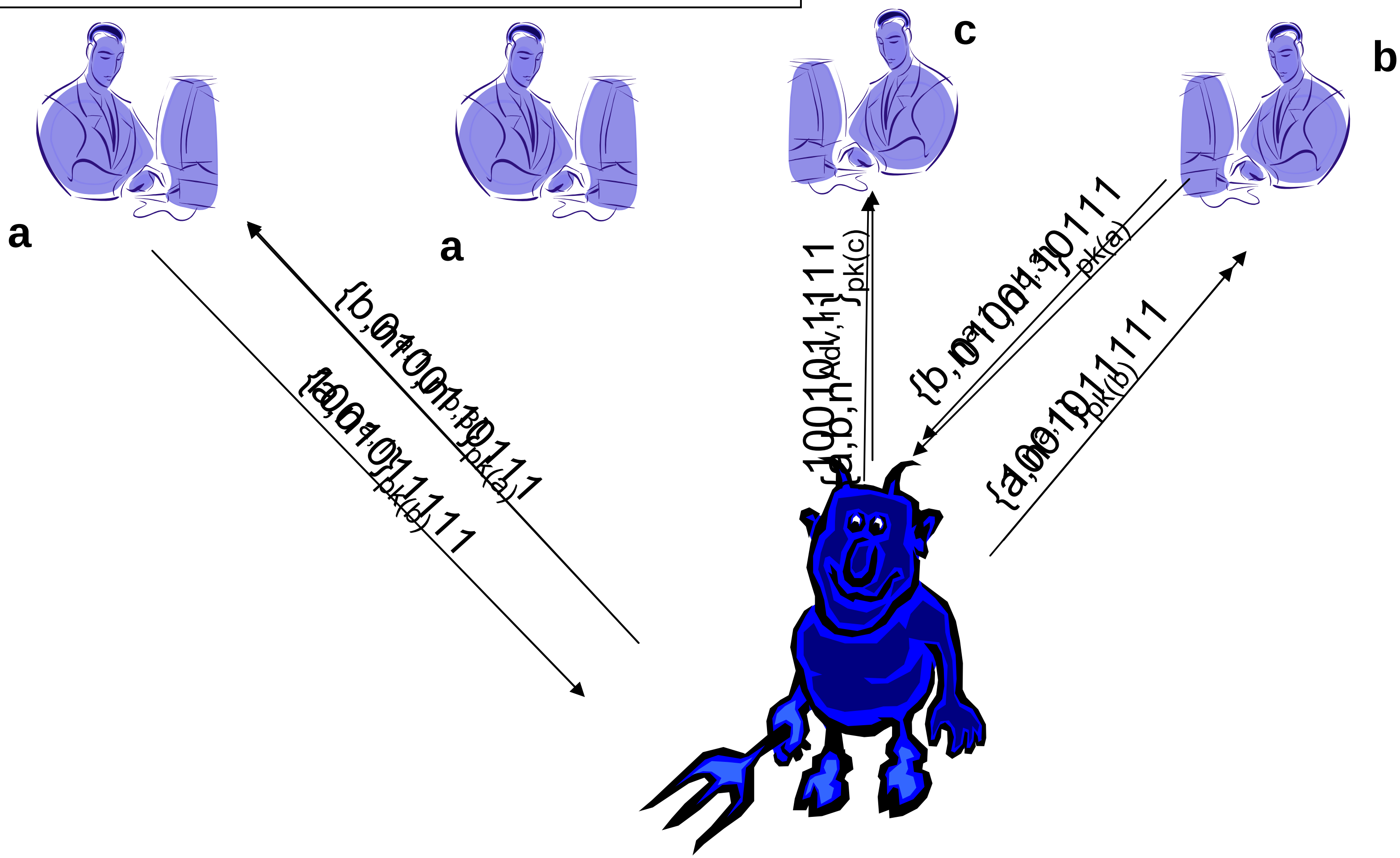
Send $\{X\}_{pk_B}$

Receive $\{A,Y\}_{pk_B}$

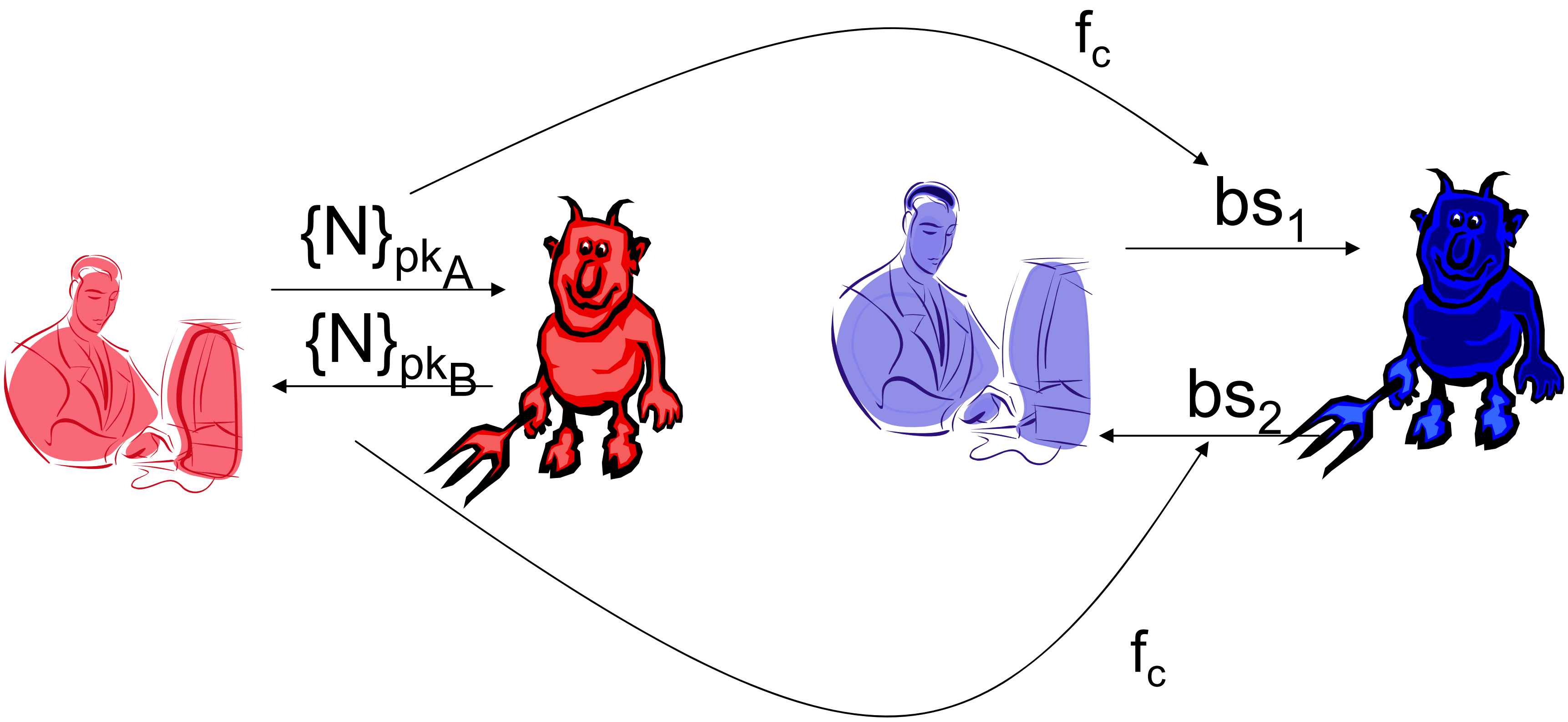
Send $\{B,Y,N_B\}_{pk_A}$

Receive $\{N_B\}_{pk_B}$

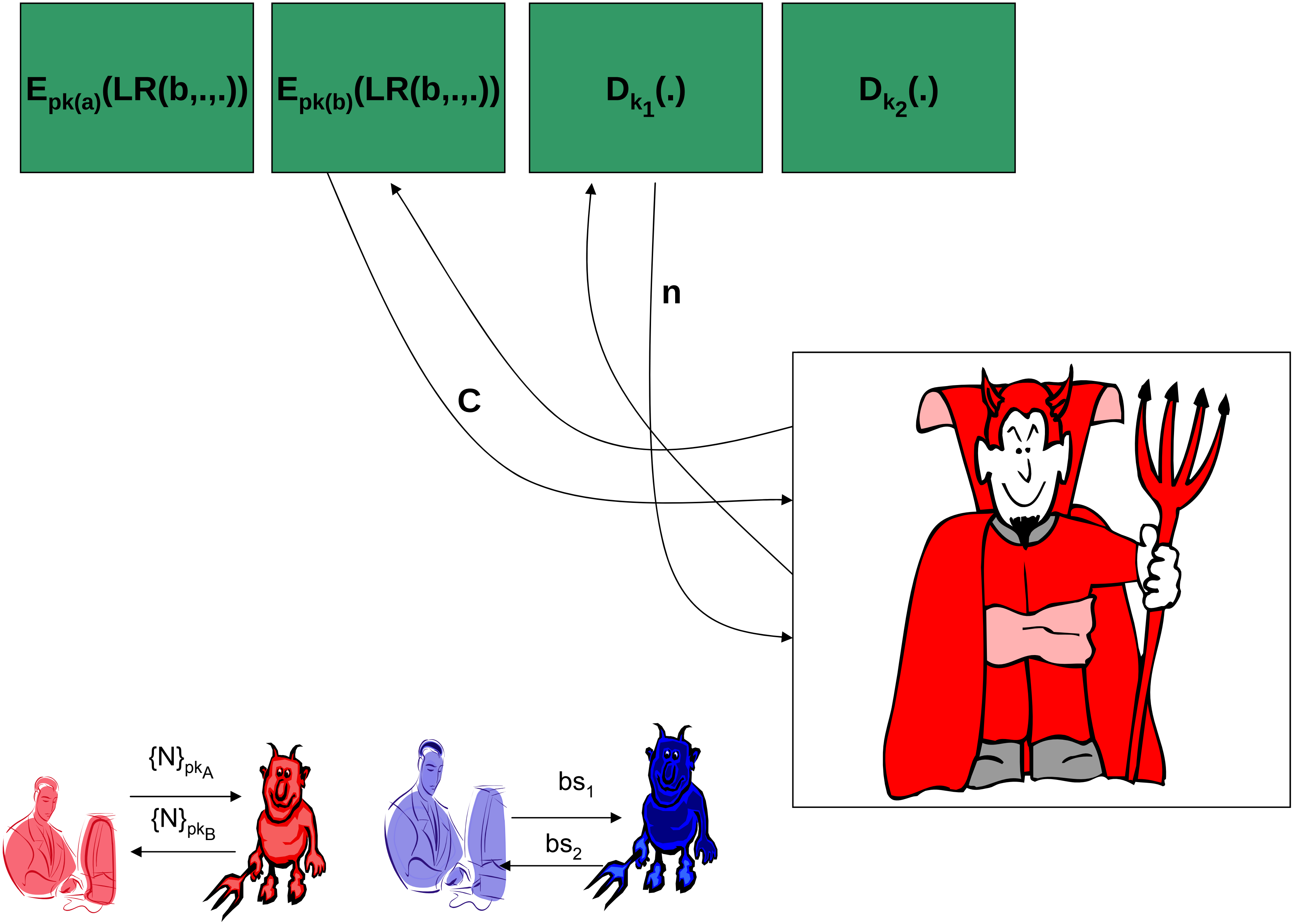
lic executions



Step 3: The symbolic trace is Dolev-Yao



Given an adversary that produces traces that are not Dolev-Yao, use that adversary to break the security of the basic primitive(s)



$E_{pk(a)}(LR(b,.,.))$
$E_{pk(b)}(LR(b,.,.))$
$D_{k_1}(.)$
$D_{k_2}(.)$
n
c
$\{N\}_{pk_A}$
$\{N\}_{pk_B}$
bs_1
bs_2

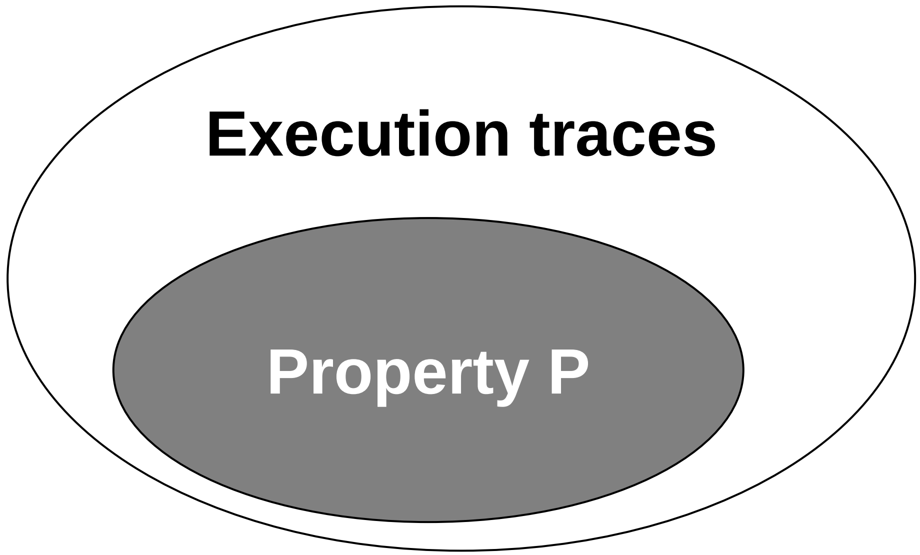
Trace mapping lemma

- Let ‟ be a protocol and *A* a computational adversary. If ‟ is implemented with secure primitives then almost all of the computational traces of ‟ are images of symbolic Dolev-Yao traces.

Prob[< B, < f_c : Tr_c(‟ ($R_‟$), A(R_A)) = f_c(Tr_s(‟ ,B))]
is overwhelming

Computational soundness for trace properties

Security properties



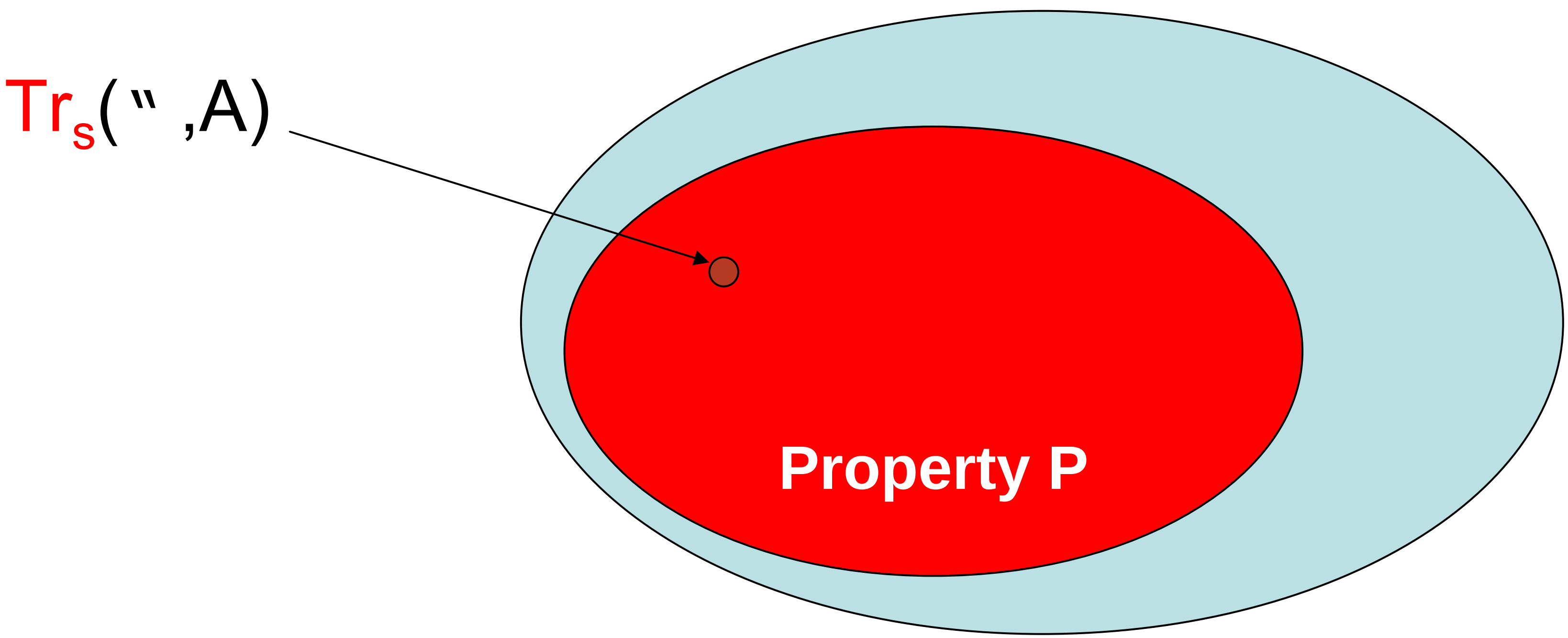
A security property is a predicate on the set of possible traces

E.g.: Matching conversations: every session of user B (with A) that finishes successfully has a matching session of user A

Security Properties - symbolically

- Protocol " satisfies security property P_s

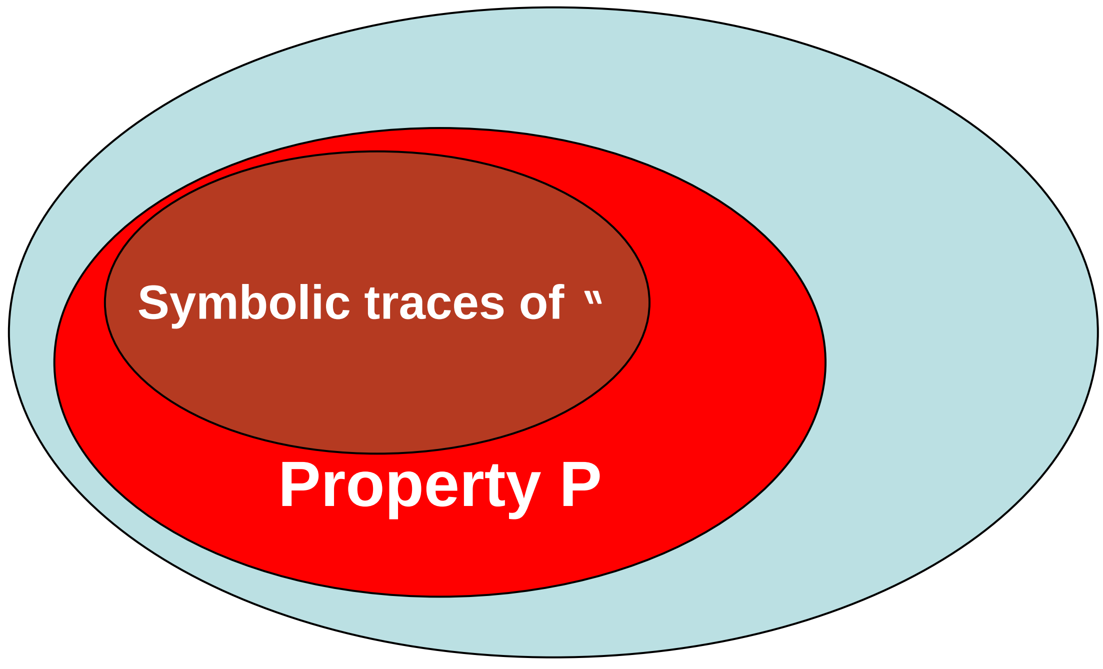
  (" ¶$_s$ P_s) iff (;A) Tr_s(",A)5 P_s



Security Properties - symbolically

- Protocol “ satisfies security property P_s

  (“ $¶_s$ P_s) iff (;A) Tr_s(“ ,A)5 P_s



Security Properties - computationally

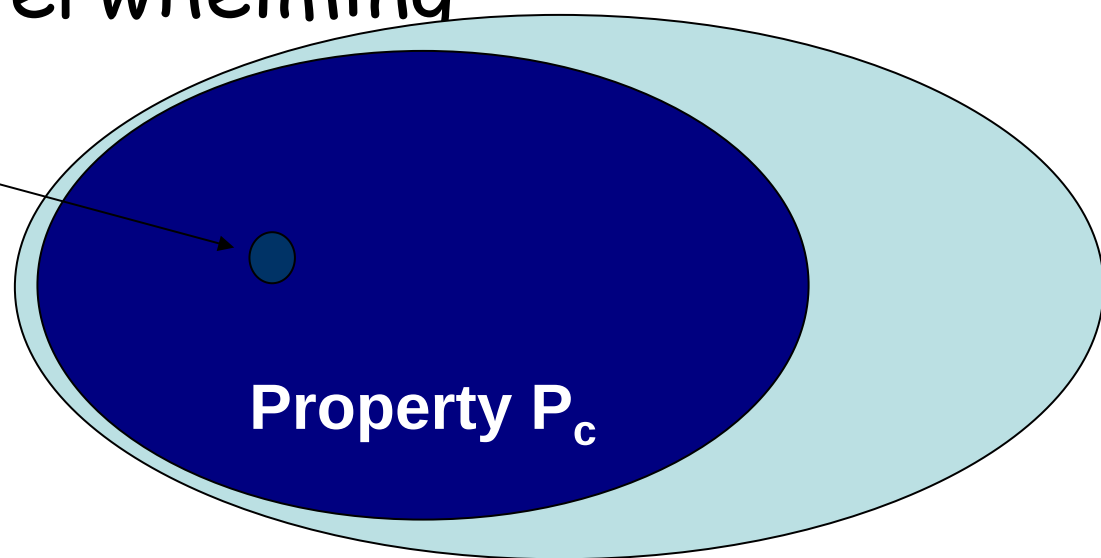
- Protocol ‟ satisfies computationally property $P_{c:}$

‟ $¶_c$ P_c iff

(;p.p.t A) Pr [Tr_c(‟ ($R_‟$), A(R_A))5 P_c]

is overwhelming

Tr_c(‟ ($R_‟$),A(R_A))



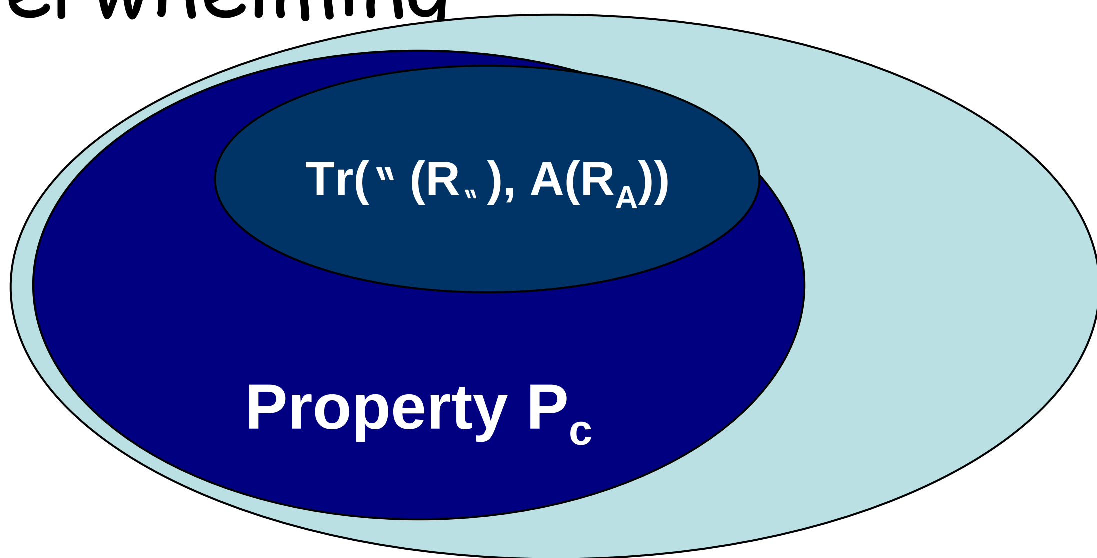
Security Properties - computationally

- Protocol ‟ satisfies computationally property P_{c}:

$$‟ \ ¶_{c} P_{c} \text{ iff}$$

$$(;\text{p.p.t } A) \ \Pr[\, Tr_{c}(‟\,(R_{‟}), A(R_{A})) 5 \ P_{c} \,]$$

is overwhelming



Translation of trace properties

Let P_s be a symbolic security property and let P_c = ″(P_s)= ;$_f$ f(P_s) (the union is after all appropriate mappings f). If the mapping lemma holds then:

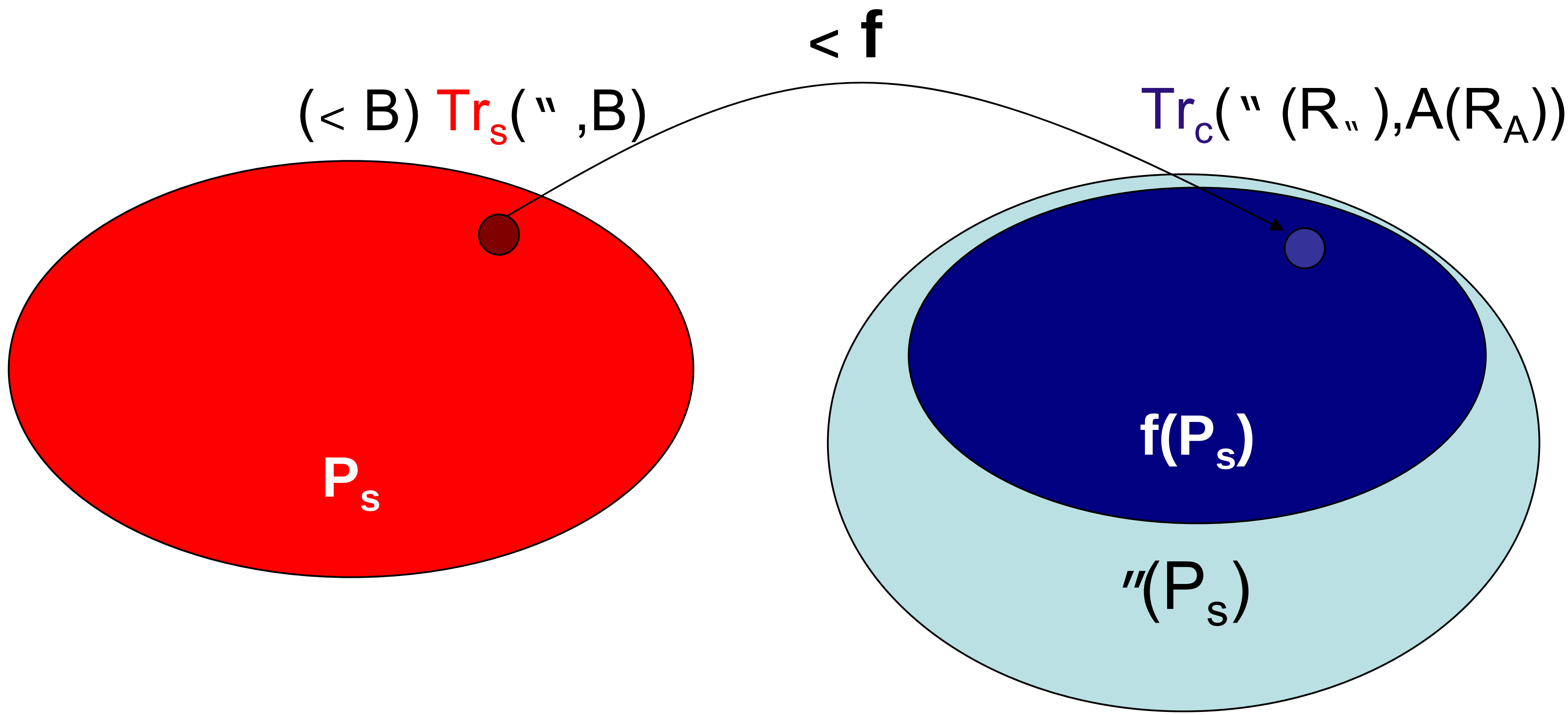
THEOREM: Let " be a protocol. Then:

" ¶$_s$ P_s , " ¶$_c$ P_c

Proof

Let ‟ be a protocol and *A* a computational adversary.

Pick R‟ and R_A. Then (with overwhelming probability):



Soundness for secrecy properties

Soundness for secrecy
[Cortier, Warinschi]

- For the N… …der Lowe protocol:

NSL ¶$_s$ Secret(N_B)

For any session t of B with an honest party A, the nonce n^t that instantiates N_B in session t is never sent by the adversary in clear over the network

Send {…

…$\{A,Y\}_{pk_B}$

Receive $\{B,N_A,X\}_{pk_A}$

Send $\{B,Y,N_B\}_{pk_A}$

Send $\{X\}_{pk_B}$

Receive $\{N_B\}_{pk_B}$

Soundness for secrecy

- The mapping lemma implies a notion of computational secrecy:
- (With overwhelming probability) the adversary cannot output any of the nonces that instantiate variable N_B in sessions of B with honest A
- …but this security notion – onewayness – is cryptographically unsatisfying

Computational secrecy

- Computational secrecy for nonce N in session t: prior to the execution select n_0, n_1. Run the protocol with n_b as value for N_B in session t. Give n_0,n_1 to the adversary and ask him to guess b
- NSL$\vDash_c$ Secret(N) if N is computationally secret in any session of B with an honest party

Soundness for secrecy

- For any protocol " implemented with secure primitives (digital signatures, public key encryption, nonces)

  " ¶$_s$ Secret(N) , " ¶$_c$ Secret(N)

- The proof relies on the computational adversary to only perform Dolev-Yao operations

Soundness for hash functions

Hash functions

[Cortier, Kremer,Küsters,Warinschi]

- The trace mapping lemma holds if hash functions are implemented by random oracles
  - Hash values can be interpreted as symbolic terms by observing the communication with the random oracle
- … soundness holds for trace properties
- How about secrecy?

Soundness for secrecy does not hold anymore

- Consider a protocol ‟ where A sends to B the message $h(N_A)$, where N_A is a random nonce. Then
- ‟ $¶_s$ Secret(N_A) is true
- ‟ ¶ Secret(N_A) is not true
- hold

Since given $h(n_b)$, n_0,n_1 the adversary can easily recover b

…but it can be recovered

- Define the pattern that the adversary can observe when *given* N. In particular:
  - $\mathrm{pattern}_N(\{N\}_{pk}) = \Box_{pk}$
  - $\mathrm{pattern}_N(h(N)) = h(N)$
  - $\mathrm{pattern}_N(h(N')) = h(\Box)$

Stronger notion of secrecy

- Stronger notion of secrecy for nonces:

  " ¶$_s$ SSecret(N) if for any instantiation $n^\dagger$ of nonce N and for any adversary A, $n^\dagger$ does not occur in $\text{pattern}_{n^\dagger}(Tr_s(\text{" },A))$
- Computational soundness for secrecy holds:

  " ¶$_s$ SSecret(N) , " ¶$_c$ Secret(N)

Additional results

Non-interactive zero-knowledge
[Backes,Unruh]

- Consider a specification language for protocols where non-interactive ZK statements can be used
- Identify the requirements needed to ensure that a mapping lemma holds
  (unpredictable non-interactive multi-theorem adaptive extraction zero-knowledge argument of knowledge with deterministic verification and extraction)
  - Extractability
  - Non-malleability
  - Unpredictability

Computational soundness for a process calculus
[Cortier, Comon-Lundh]

- Protocols written in a subset of applied `-calculus
  - Use symmetric key-encryption
- Define symbolic and computational executions for processes
- Soundness of observational equivalence: processes indistinguishable, symbolically, are indistinguishable by a computational attacker.

Commitment schemes
[Galindo, Garcia, van Rosum]

- Soundness for non-malleable commitments
- Commitments are similar to encryption

Some observations

Extractability

- Needed for interpreting uniquely each bitstring as a term
- Is ensured by either cryptographic security (e.g. integrity of encryption, collision resistance for hashes, extractability for ZK, message revealing signatures), extra randomization, and/or tagging of messages with types

Executability (simulatability)

- Needed to ensure that the execution of the protocol can be simulated for the adversary
- Identify appropriate restrictions on the protocols to ensure execution is possible (at the very least “normal” executability but possibly more)

Non-malleability

- Usually symbolic axiomatization implies non-malleability
- The Lowe-type attack on the NS implementation with IND-CPA scheme is permitted by non-malleability
- Seems to be a (the) useful property (soundness for non-malleable commitments and ZK)

Some future directions

- Compositional soundness results
- Convincing applications
- Relevance to actual implementations

Thank you.