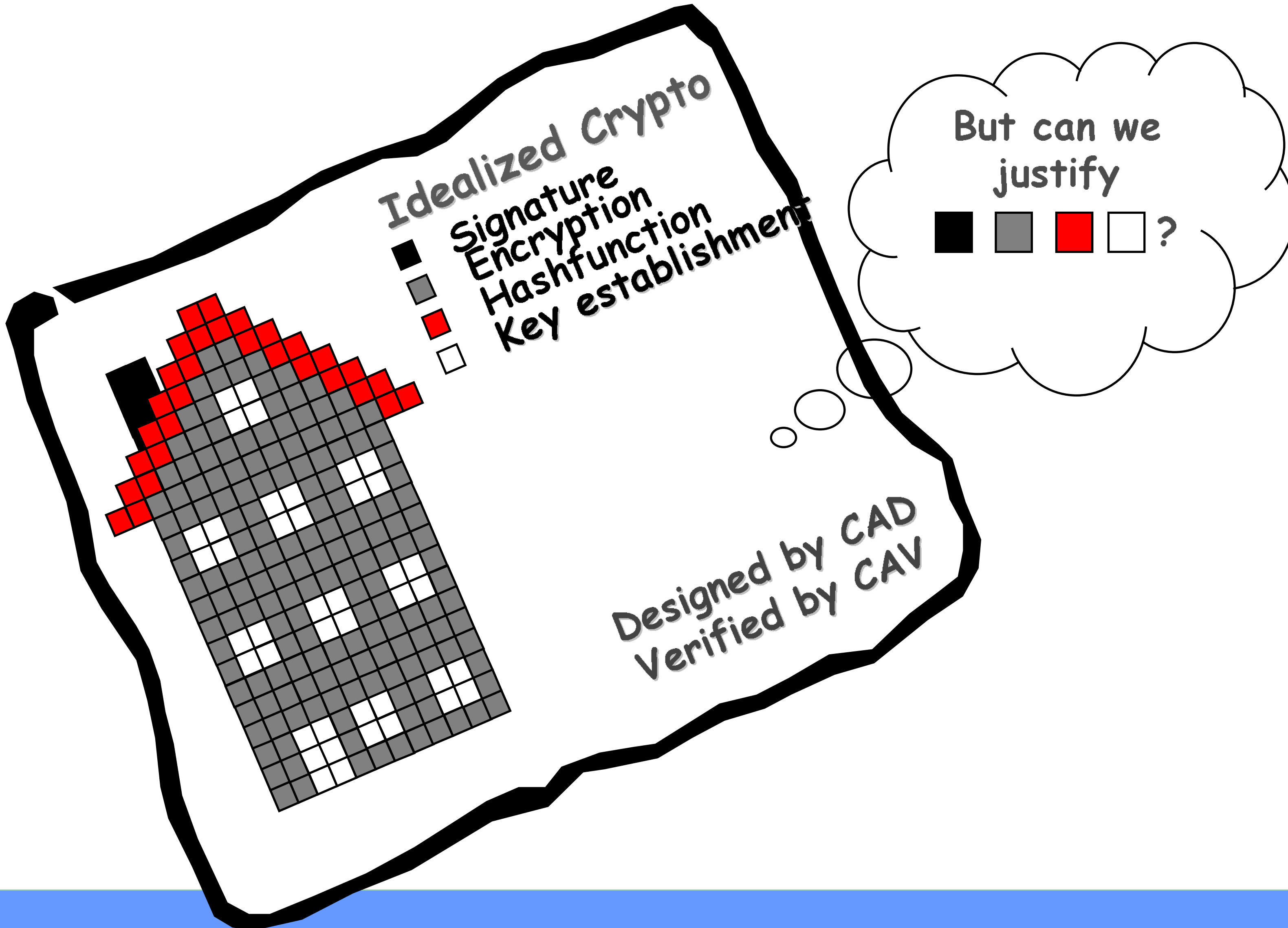
Michael Backes
Saarland University, Germany
joint work with Birgit Pfitzmann and Michael Waidner

Secure Reactive Systems, Day 3:

Reactive Simulatability – Property Preservation and Crypto. Examples

Tartu, 03/01/06

Recall the Big Picture

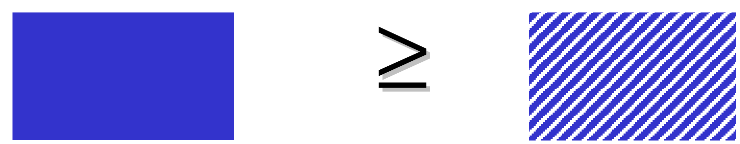


Recall the RS Framework

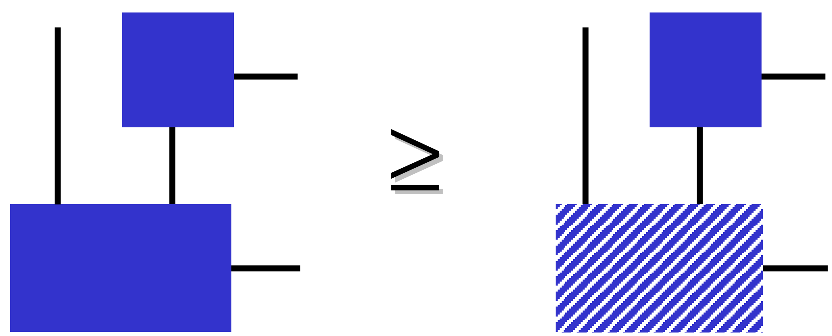
- **Precise system model allowing cryptographic and abstract operations**
- **Reactive simulatability with composition theorem**
- Preservation theorems for security properties
- **Concrete pairs of idealizations and secure realizations**
- Sound symbolic abstractions (Dolev-Yao models) that are suitable for tool support
- Sound security proofs of security protocols: NSL, Otway-Rees, iKP, etc.
- **Detailed Proofs** (**Poly-time**, cryptographic bisimulations with static information flow analysis, …)

Composition – One System

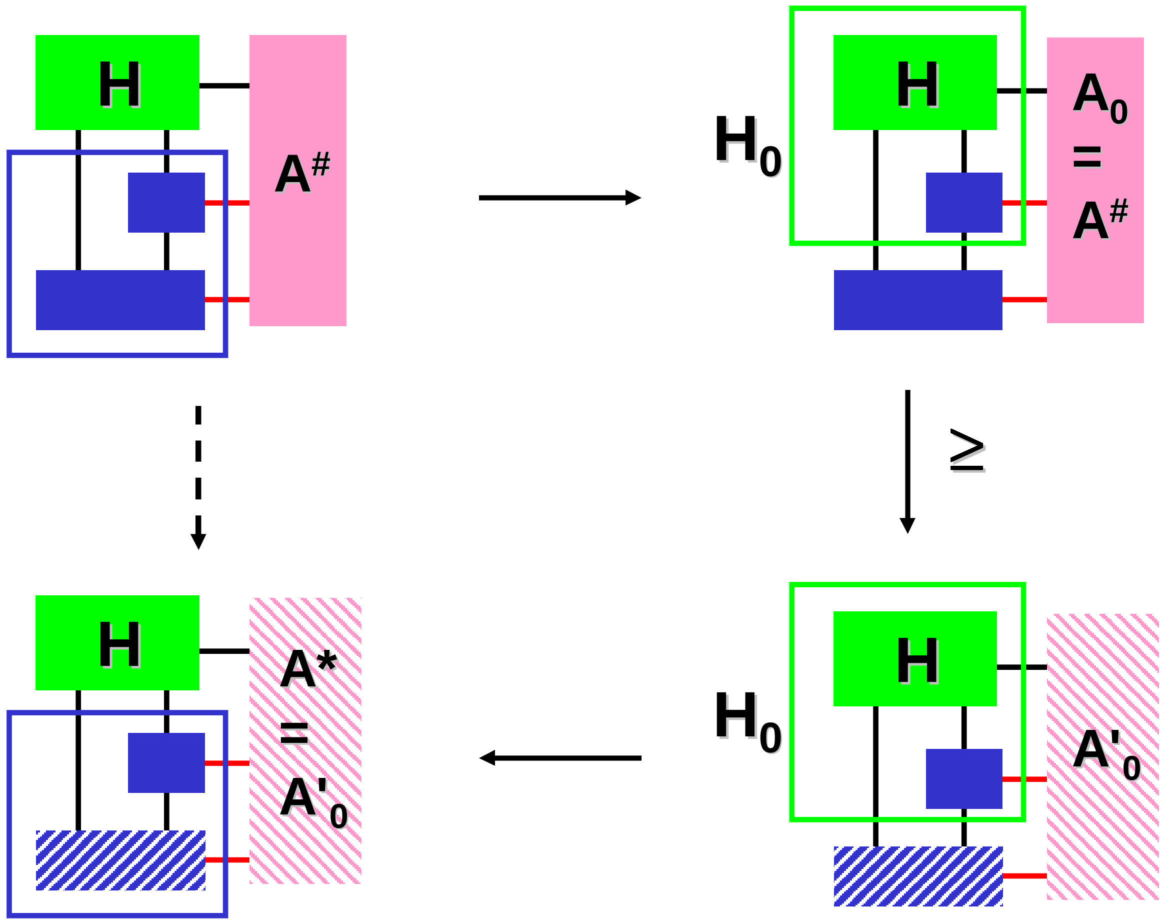
Given:



Then this holds:



Proof Idea (Single Composition)



Recall the RS Framework

- Precise system model allowing cryptographic and abstract operations
- Reactive simulatability with composition theorem
- **Preservation theorems for security properties**
- **Concrete pairs of idealizations and secure realizations**
- Sound symbolic abstractions (Dolev-Yao models) that are suitable for tool support
- Sound security proofs of security protocols: NSL, Otway-Rees, iKP, etc.
- **Detailed Proofs** (**Poly-time**, cryptographic bisimulations with static information flow analysis, …)

Abstraction of one-step Public-Key Encryption

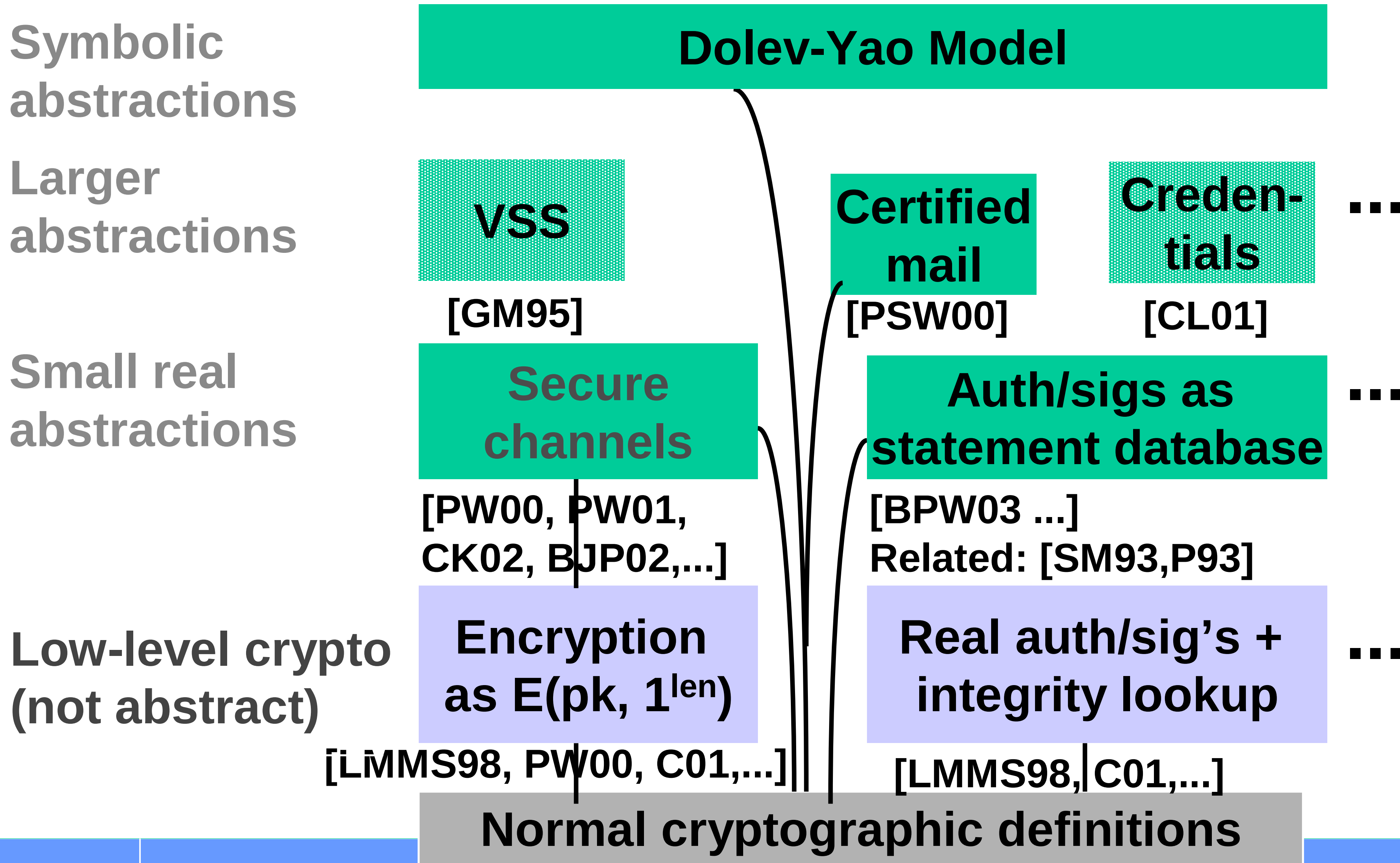
- **On the board…**

Example: Encryption, passive

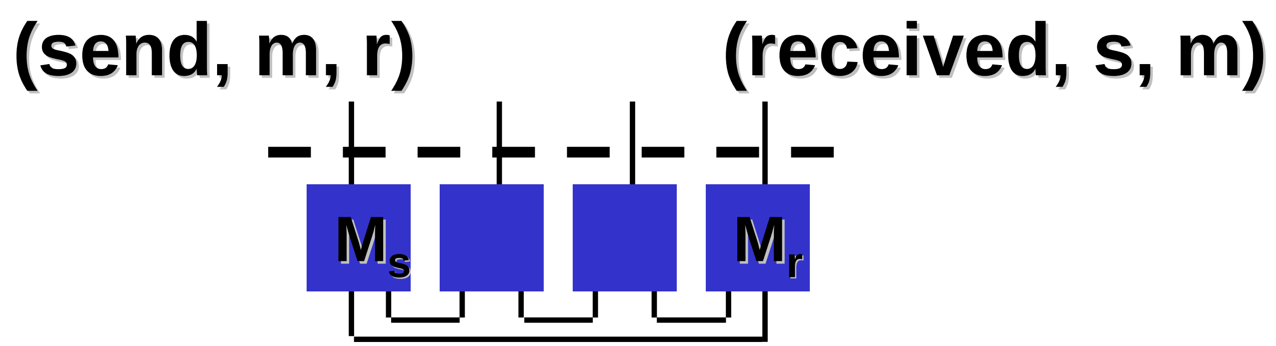
$\forall A_1, A_2 \in PPT$:

$P(b^* = b ::$	(Attacker success)
$(sk, pk) \leftarrow gen(k);$	(Keys)
$(m_0, m_1, v) \leftarrow A_1(k, pk);$	(Message choice)
$b \in_R \{0, 1\};$	
$c := enc(pk, m_b);$	(Encrypt)
$b^* \leftarrow A_2(v, c)\ )$	(Guess)
$\leq 1/2 + 1/\text{poly}(k)$	(Negligible)

Cryptographic Idealization Layers



Real System



in_s: (send, m, r): $enc_r(sign_s(s, m, r))$

$net_{r,s}$: ($enc_r(sign_{s,c}(s, m, r))$:

1. Decrypt, check signature, s, r → abort at failure
2. Output (received, s, m)

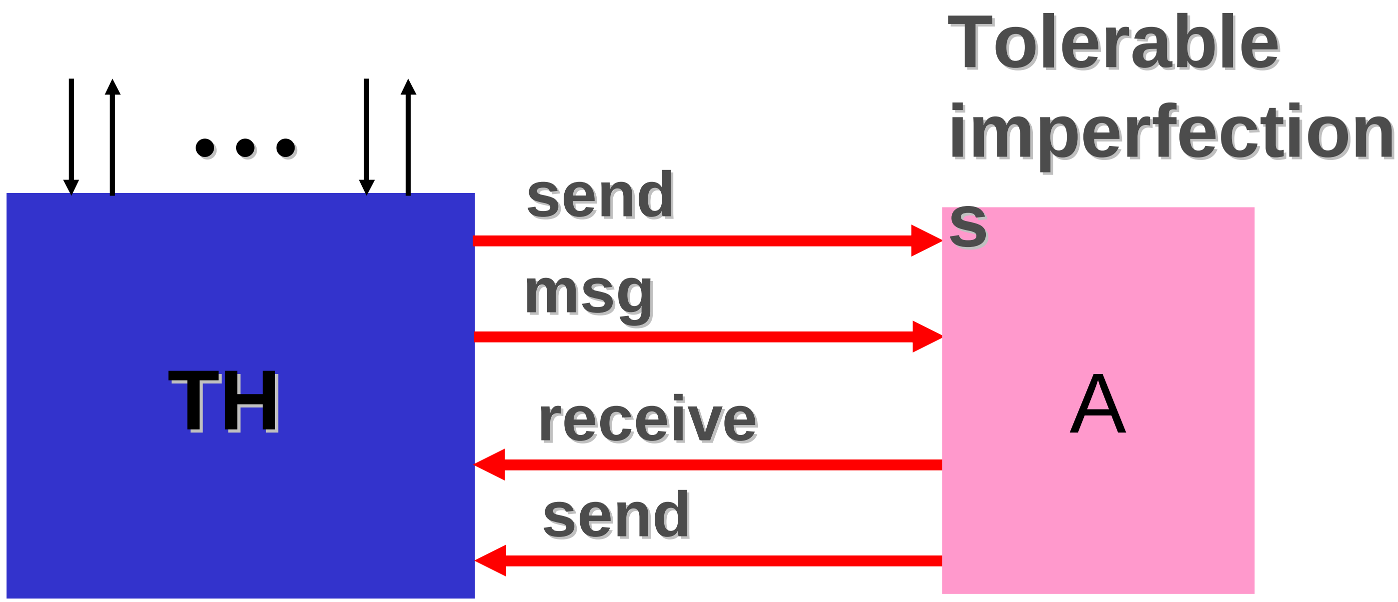
Recall Naive Approach

E.g., secure channel

Not a good abstraction since not enough information for the simulator:

- Who is sender? Who is recipient?
- Length of m?
- No availability …

Better Abstraction



in$_s$: (send, m, r):

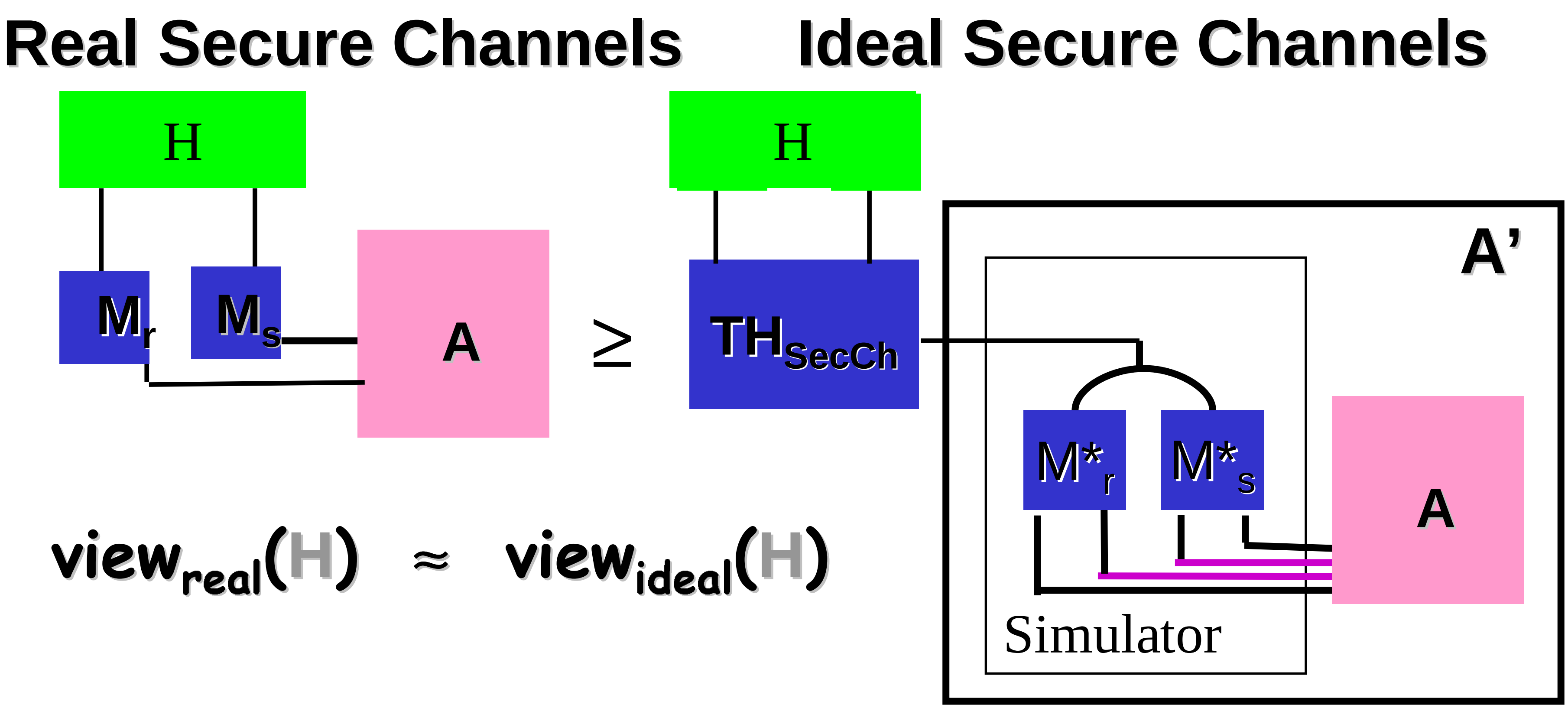
msg$_{s,r}$:= msg$_{s,r}$ & m,

output (i, l, s, r) to Adversary

from_adv$_r$: (send,i,s):

m:= msg$_{s,r}$ [i], output (received, s, m)

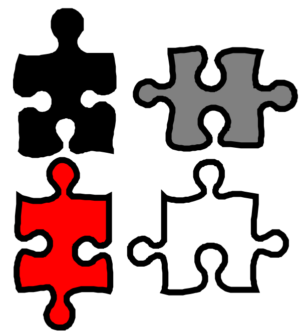
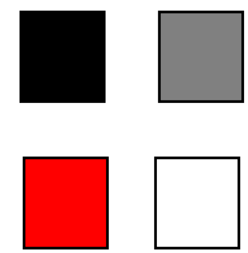
Proof Idea



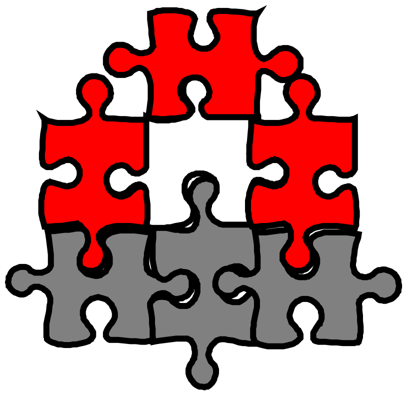
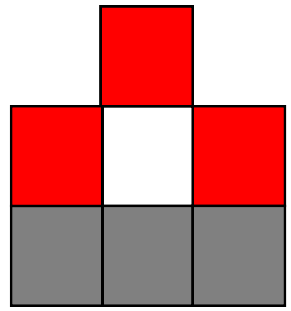
1. Proof by probabilistic bisimulation possible for „most“ cases
2. Collect remaining traces in error sets (e.g., for forged signatures)
3. Show reduction proof of error sets against underlying crypto-primitive (e.g., against security of the signature scheme)

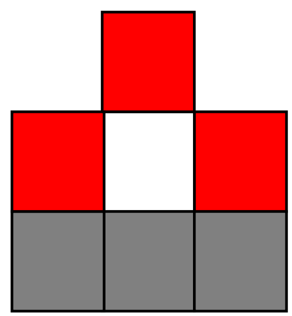
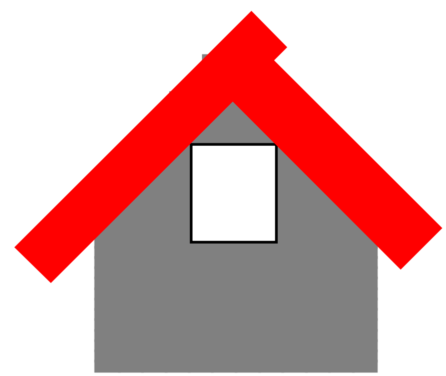
Explicit Security Requirements in the Model

Recall Prior Result

- **“as secure as” (reactive simulatability)**
- **for certain versions of**  **and** 

Specification Styles

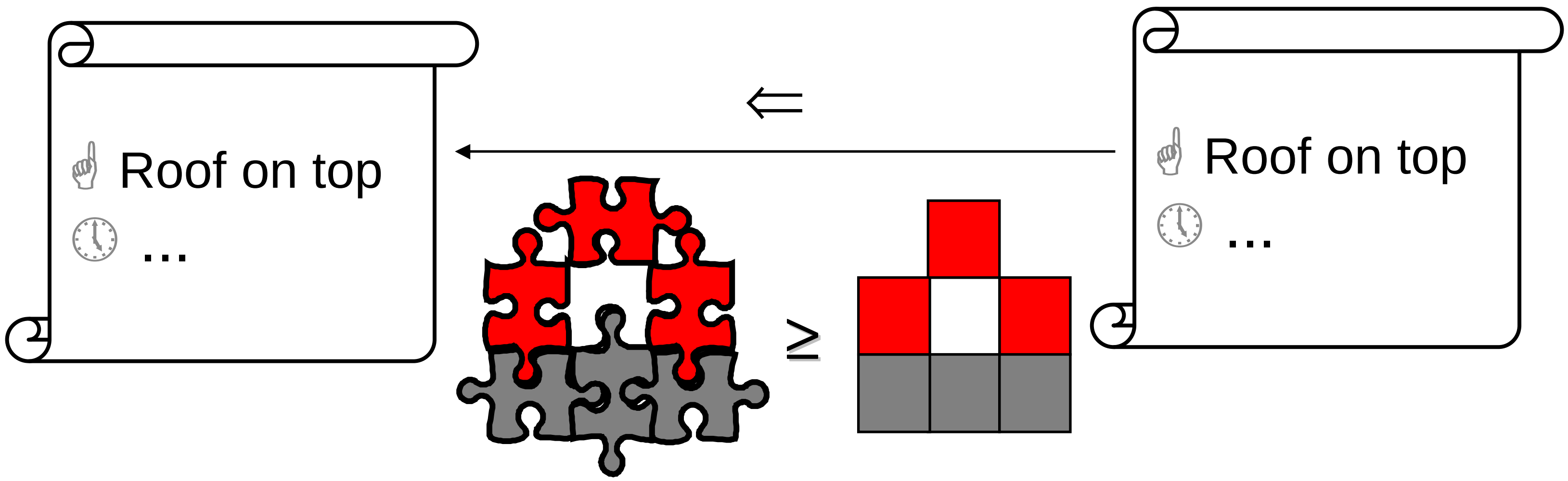
- **Is**  ≥  **what people want?**
- **Often yes, in particular together with**

 ≥ 

  - **E.g., secure channels (see also spi calculus), certified mail**
- **But not always ...**

Alternative: Property-based spec.

- **E.g., “I want a tight roof on top”:** **integrity**
  - **Preserved by “≥”:**



Characterization

Integrity (e.g., temporal logic)

Privacy (e.g., information flow, non-interference)

Liveness: (Something good eventually happens)

- **Termination**
- **Starvation freedom**
- **Guaranteed service**

Integrity

Integrity

Abstract formulation: e.g., temporal logic over the interface of a system (ports to the user)

Cryptographic semantics: For all with linear-time semantics (set of permitted traces)

Example: "If m is input at p? at time t, then there exists a future time s such that m is output at port q!" (≈ Reliability)

A trace tr is contained in Req if

$\forall t$: t: p?m → $\exists$ s > t: s: q!m

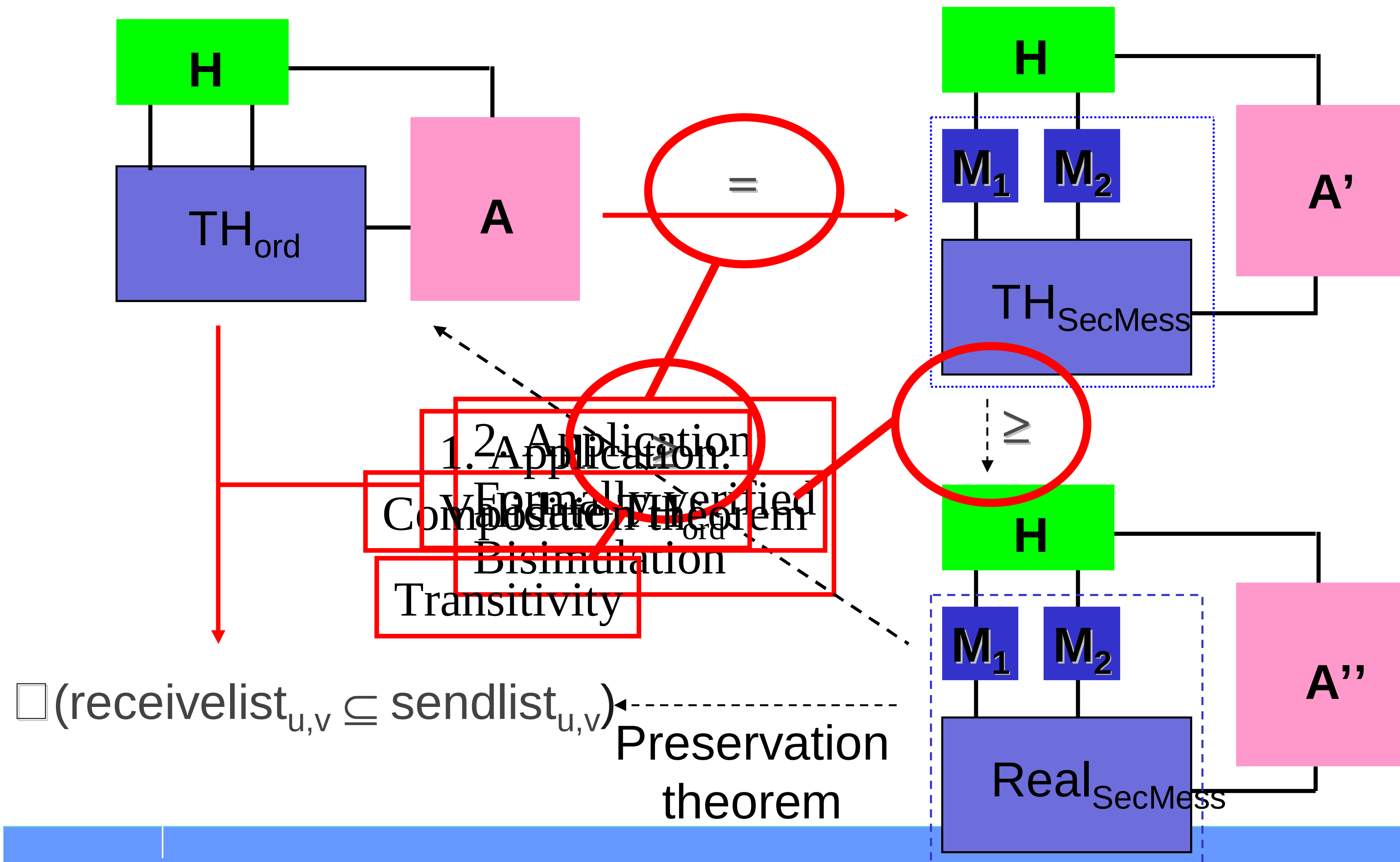
Fulfillment of Integrity

Different kinds of fulfillment:

- Perfect: **Requirement always holds**
- Computational: **For polynomial-time adversary and users only and up to negligible error probability**

Integrity Preservation Theorem: **Simulatability preserves "≥": $Sys_1 \geq Sys_2$ and Sys_2 |= Req implies Sys_1 $|=^{poly}$ Req**

Example: Ordered Secure Channels over Unordered Ones



Cryptographic Non-Interference (Transitive)

Privacy

- **No single well-established type of privacy properties in formal methods**
- **Most common type here: Non-interference**
- **Lots of application areas:**
  - **Secure operating systems [De76,De77]**
  - **Confinement: trusted program leaks information through covert channels**
  - **Renewed importance with extensible systems: applets, kernel extensions, mobile agents, etc.**

Some Prior Approaches

Non-probabilistic Reactive systems: [Many]

- **Based on process calculi**
- **Definitions are the main issue, different types of non-interference.**
- **Main problem here: refinement**

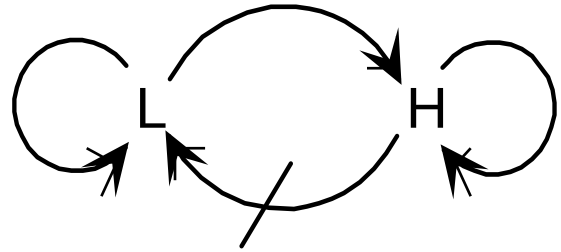
Probabilistic Reactive systems [Gr92]

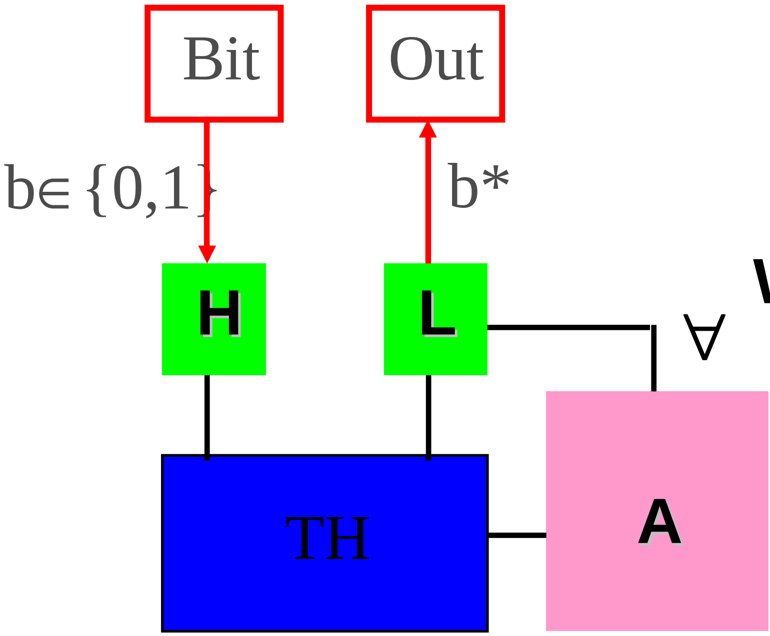
- **Gray‘s definition „Probabilistic Non-Interference“ stands out**
  - **For all high-level environment behaviours same probability distribution of the low-events.**
  - **Perfect fulfillment only, not yet suited for real cryptography → introduce error probabilities, etc.**

Prior work (cont'd)

	Deterministic	Non-deterministic	Probabilistic	Crypto-graphic
Non-Interference	**GM 82**	**Many**	**Gray 92**	New

Cryptographic Non-Interference





Want to express: No information can flow from H to L

Idea: Whatever H does, L will not recognize it

$P(b=b^*) \leq 1/2 + Negl$

- **Now error probabilities, computational restrictions**
- **„Guessing a bit“ is a typical concept in cryptography**
  → Closely related to cryptographic definitions

Preservation under Simulatability

- **Preservation Theorem (Informal):**

  Whenever an abstractions fulfills a cryptographic non-interference requirement, then every secure implementation of it also fulfills this requirement.

- **Formally:**

  $\mathbf{Sys_1 \geq Sys_2 \;\wedge\; Sys_2 \models NIReq_{H\_L} \Rightarrow Sys_1 \models NIReq_{H\_L}}$

Cryptographic Non-Interference (Intransitive)

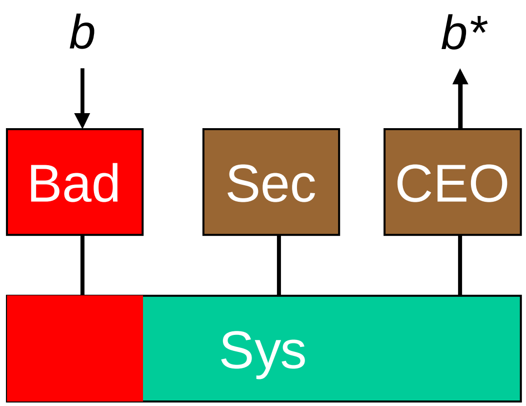
A Scenario for Intransitive Non-Interference

Prior work (cont'd)

	Deterministic	Non-deterministic	Probabilistic	Crypto-graphic
Non-Interference	**GM 82**	**Many**	**Gray 92**	New
Intransitive	**GM 84**	**Rushby 92, Pinsky 95, RG 99, SRS+ 00**	New	New

Definition 1: Blocking Non-Interference

Secretary can prevent the flow



$\forall$ Bad $\forall$ CEO $\exists$ Sec: Bad $\not\rightarrow$ CEO

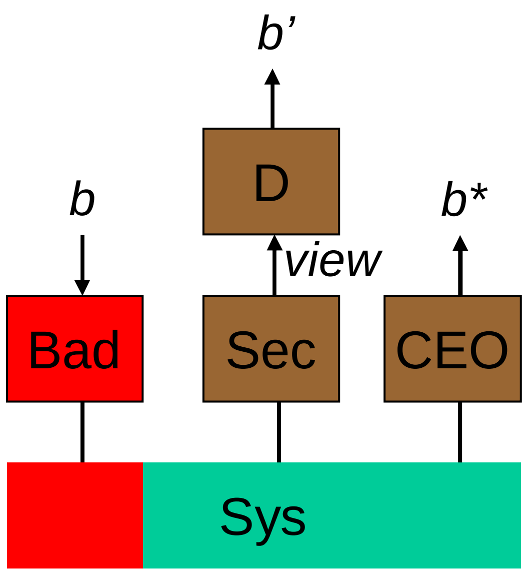
all poly-time

$$\text{Prob}(b^* = b :: r \leftarrow \text{run}_{conf};\ b := r\lceil_{\text{b\_in}} \ldots;\ b^* := r\lceil_{\text{b\_out}})$$

$$\leq ½ + \varepsilon \begin{cases} 0 \\ \text{Small} \\ \text{Negl} \end{cases}$$

Definition 2: Recognition Non-interference

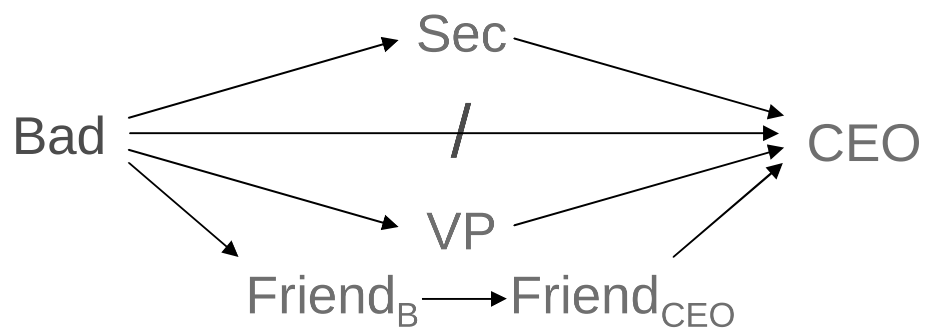
Secretary sees what's going on



CEO gets b $\Rightarrow$ Sec gets b.

$\forall$ Bad $\forall$ CEO $\forall$ Sec $\exists$ D

Arbitrary Flow Graphs

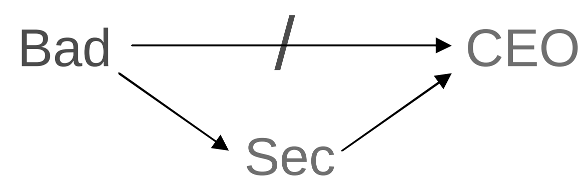


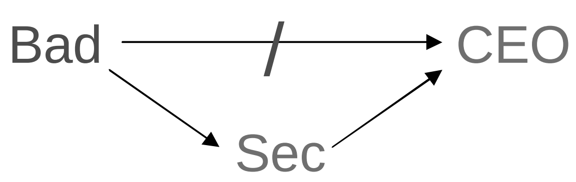
∀ Bad ∀ CEO ∀ cuts ∃ Cut-Distinguisher

Preservation under Simulatability

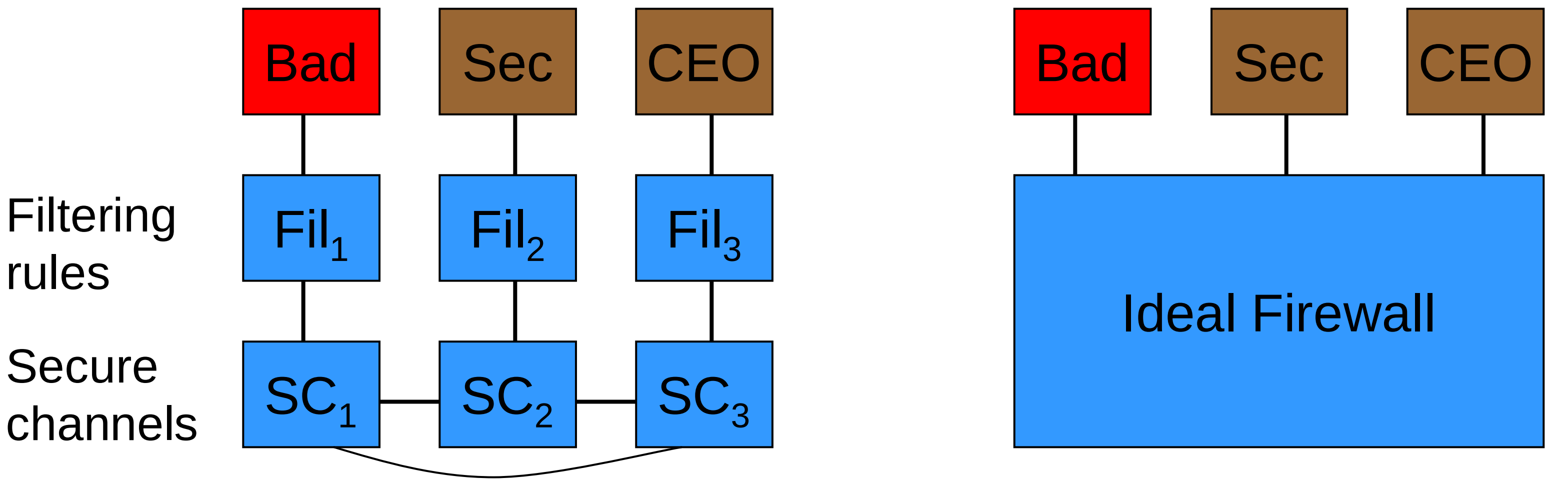
Theorem:

Sys $\geq^{sec}$ IdealSys





Implementation with Cryptographic Firewall



Prove recognition NI

Michael Backes
Saarland University, Germany
joint work with Birgit Pfitzmann and Michael Waidner

Secure Reactive Systems, Day 4:

Justifying Symbolic Abstractions of Cryptography

Tartu, 03/02/06

Recall the RS Framework

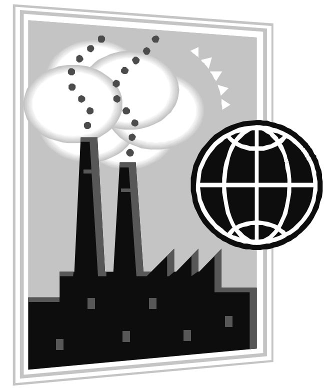
- **Precise system model allowing cryptographic and abstract operations**
- **Reactive simulatability with composition theorem**
- **Preservation theorems for security properties**
- **Concrete pairs of idealizations and secure realizations**
- **Sound symbolic abstractions (Dolev-Yao models) that are suitable for tool support**
- **Sound security proofs of security protocols: NSL, Otway-Rees, iKP, etc.**
- **Detailed Proofs (Poly-time, cryptographic bisimulations with static information flow analysis, …)**

Recall the RS Framework

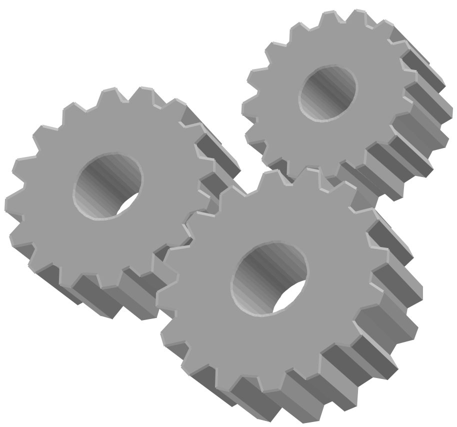
- Precise system model allowing cryptographic and abstract operations
- Reactive simulatability with composition theorem
- Preservation theorems for security properties
- Concrete pairs of idealizations and secure realizations
- **Sound symbolic abstractions (Dolev-Yao models) that are suitable for tool support**
- Sound security proofs of security protocols: NSL, Otway-Rees, iKP, etc.
- Detailed Proofs (Poly-time, cryptographic bisimulations with static information flow analysis, …)

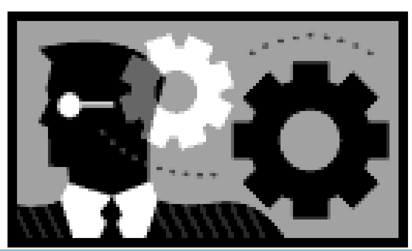
Automatic Proofs of Security

Why Formal Methods?

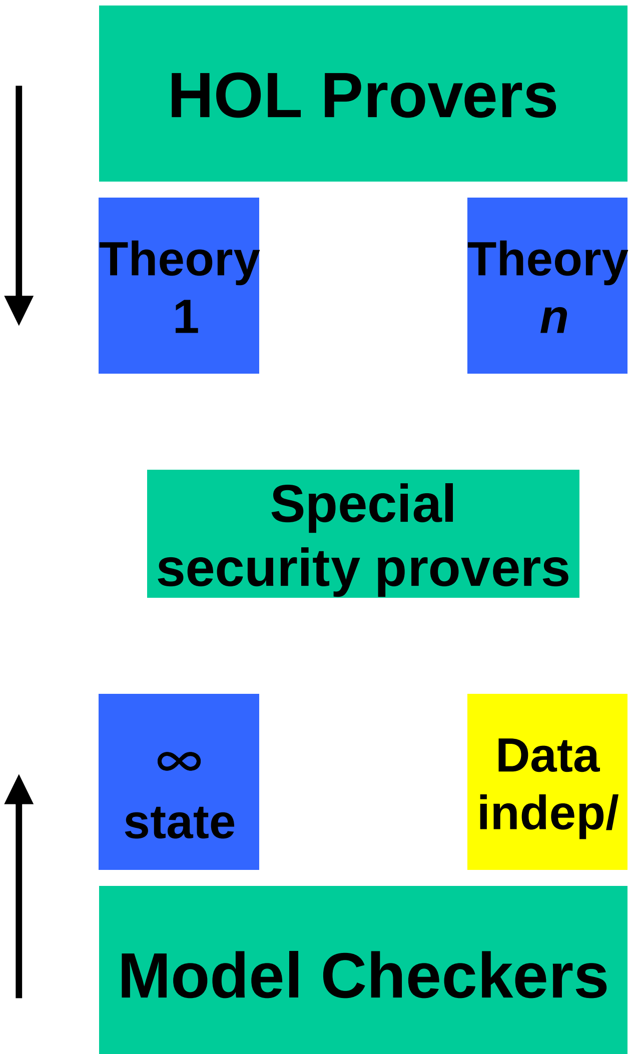


- **Automation if**
  - **Repetitive**
  - **Tedious**
  - **Prone to human errors**
  - **Critical application**
- **A top candidate: Distributed protocols**
- **Security variants for 20 years**





Protocol Proof Tools



- **Almost anything**
- **Much human interaction**

- **Special logic fragments for security**
- **Approximations: correct, not complete**

- **Fully automatic**
- **State exploration**

Automating Security Protocol Proofs

- **Even simple protocol classes & properties undecidable**
  - **Robust protocol design helps**
- **Full arithmetic is out**
- **Probability theory just developing**

So how do current tools handle cryptography?

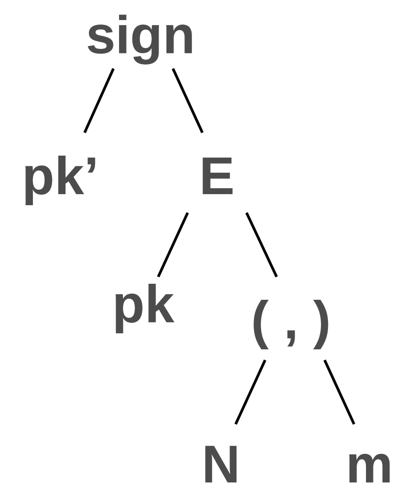
Dolev-Yao Model

- Idea [DY81]
  - Abstraction as term algebras, e.g., $D_x(E_x(E_x(m)))$
  - Cancellation rules, e.g., $D_x E_x = \varepsilon$
- Well-developed proof theories
  - Abstract data types
  - Equational 1st-order logic
- Important for security proofs:
  - Inequalities! (Everything that cannot be derived.)
  - Known as "initial model"

Important goal: Justify or replace

Dolev-Yao Model – Variants [Ours]

- **Operators and equations** **[EG82, M83, EGS85 ...]**
  - **pub enc, sym enc, nonce, payload, pairing, sigs, ...**
  - **Inequalities assumed across operators!**
- **Untyped or typed**
- **Destructors explicit or implicit**
- **Abstraction from probabilism**
  - **Finite selection, counting, multisets**
- **Surrounding protocol language**
  - **Special-purpose, CSP, pi calculus, ... [any]**



The BPW Model
(Ideal Dolev-Yao Style Library)

Dolev-Yao-style Crypto Abstractions

- **Recall: Term algebra, inequalities**
- **Major tasks:**
  - **Represent ideal and real library in the same way to higher protocols**
  - **Prevent honest users from stupidity with real crypto objects, but don't restrict adversary**
    - **E.g., sending a bitstring that's almost a signature**
  - **What imperfections are tolerable / must be allowed?**

Ideal Cryptographic Library

U

V

No crypto outputs!
Deterministic!

Commands,
payloads,
~~terms?~~ **handles**

Payloads / test results,
~~terms?~~ **handles**

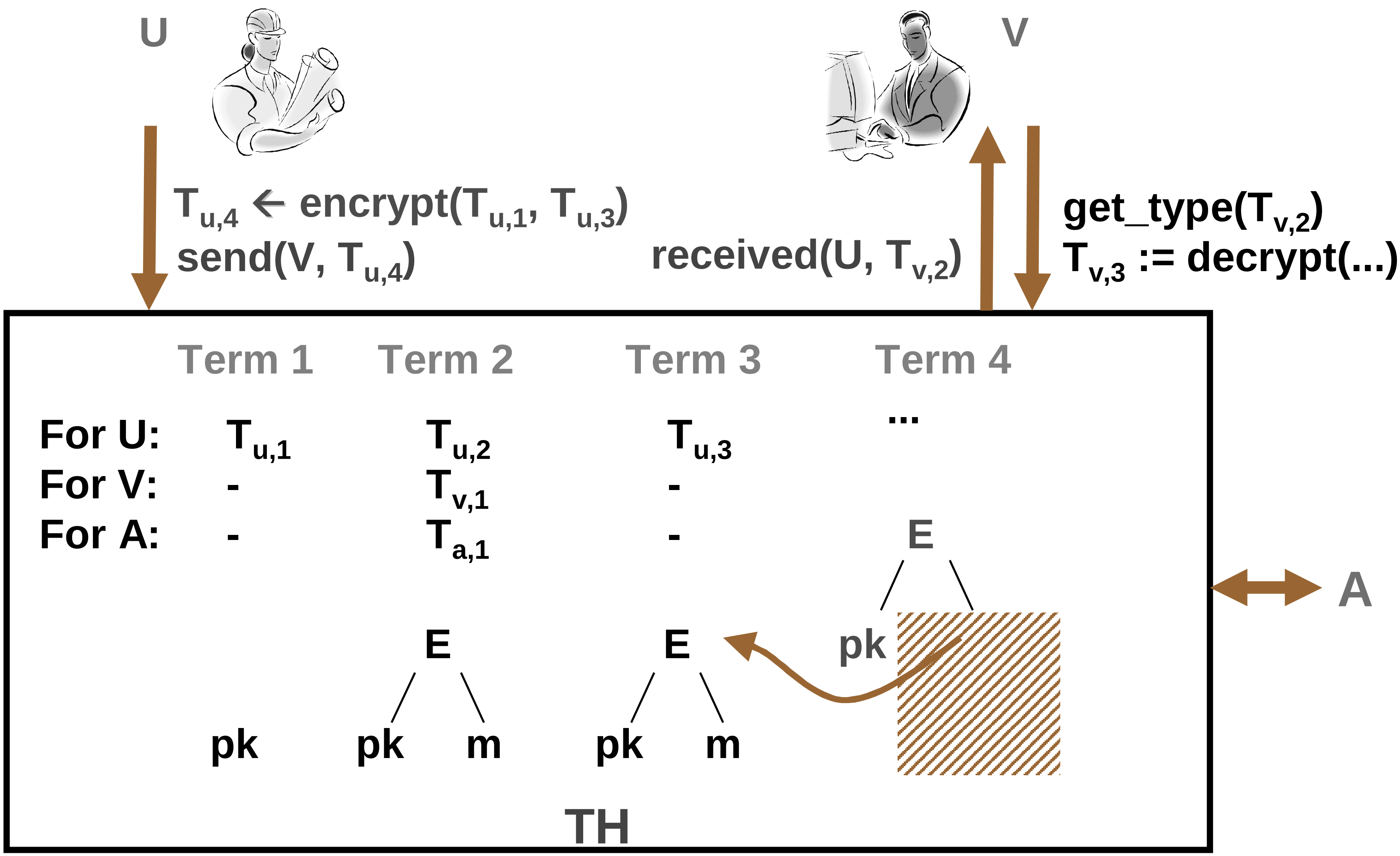
	Term 1	Term 2	Term 3	Not globally known
For U:	$T_{u,1}$	$T_{u,2}$	$T_{u,3}$	
For V:	-	$T_{v,1}$	-	
For A:	-	$T_{a,1}$	-	

	E	E
pk	pk m	pk m

TH

A

Ideal Cryptographic Library (2)



Main Differences to Dolev-Yao

Tolerable imperfections:

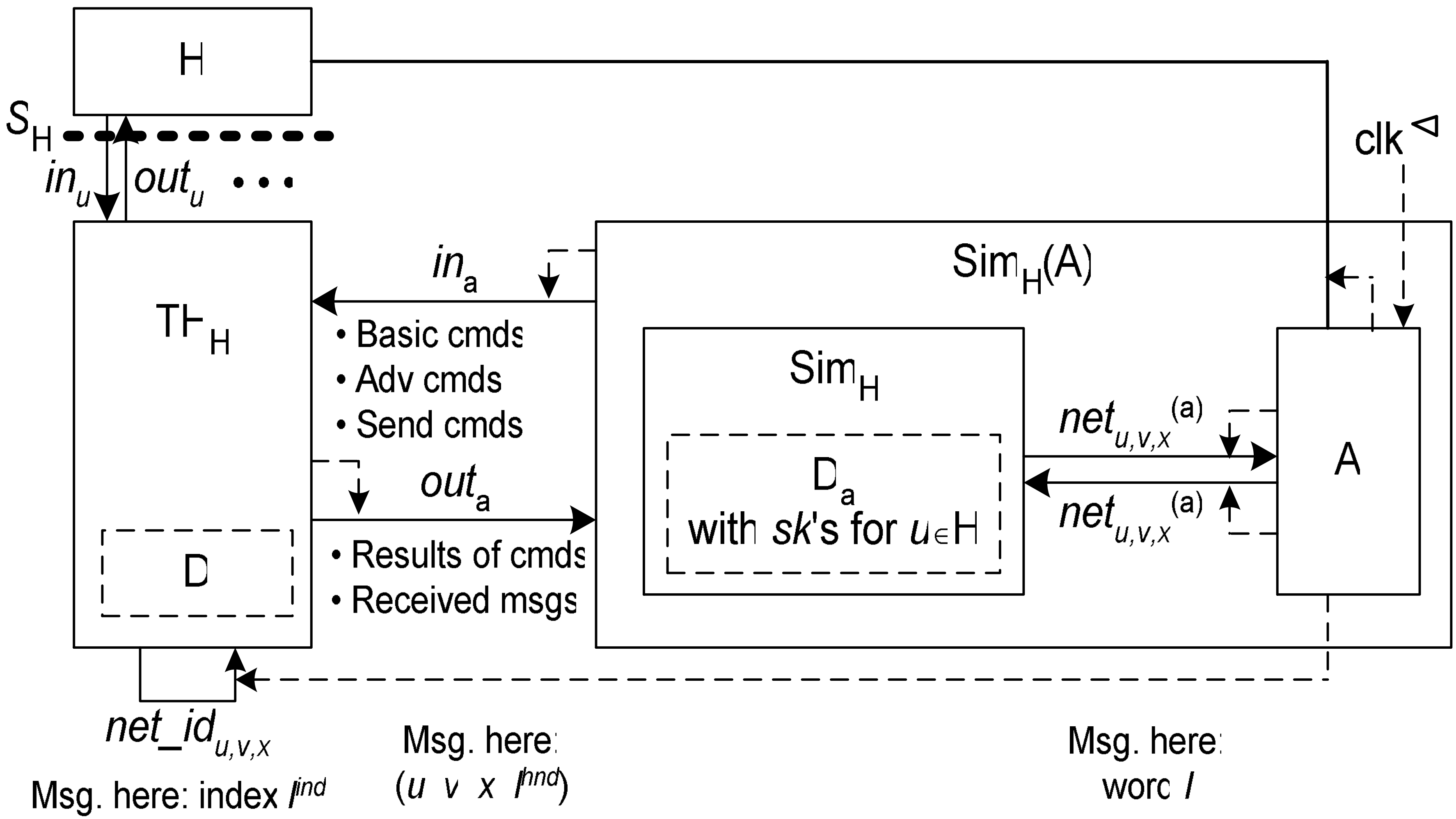
- **Lengths of encrypted messages cannot be kept secret**
- **Adversary may include incorrect messages inside encryptions**
- **Signature schemes can have memory**
- **Slightly restricted key usage for symmetric encryption**

Most imperfections avoidable for more restricted cases

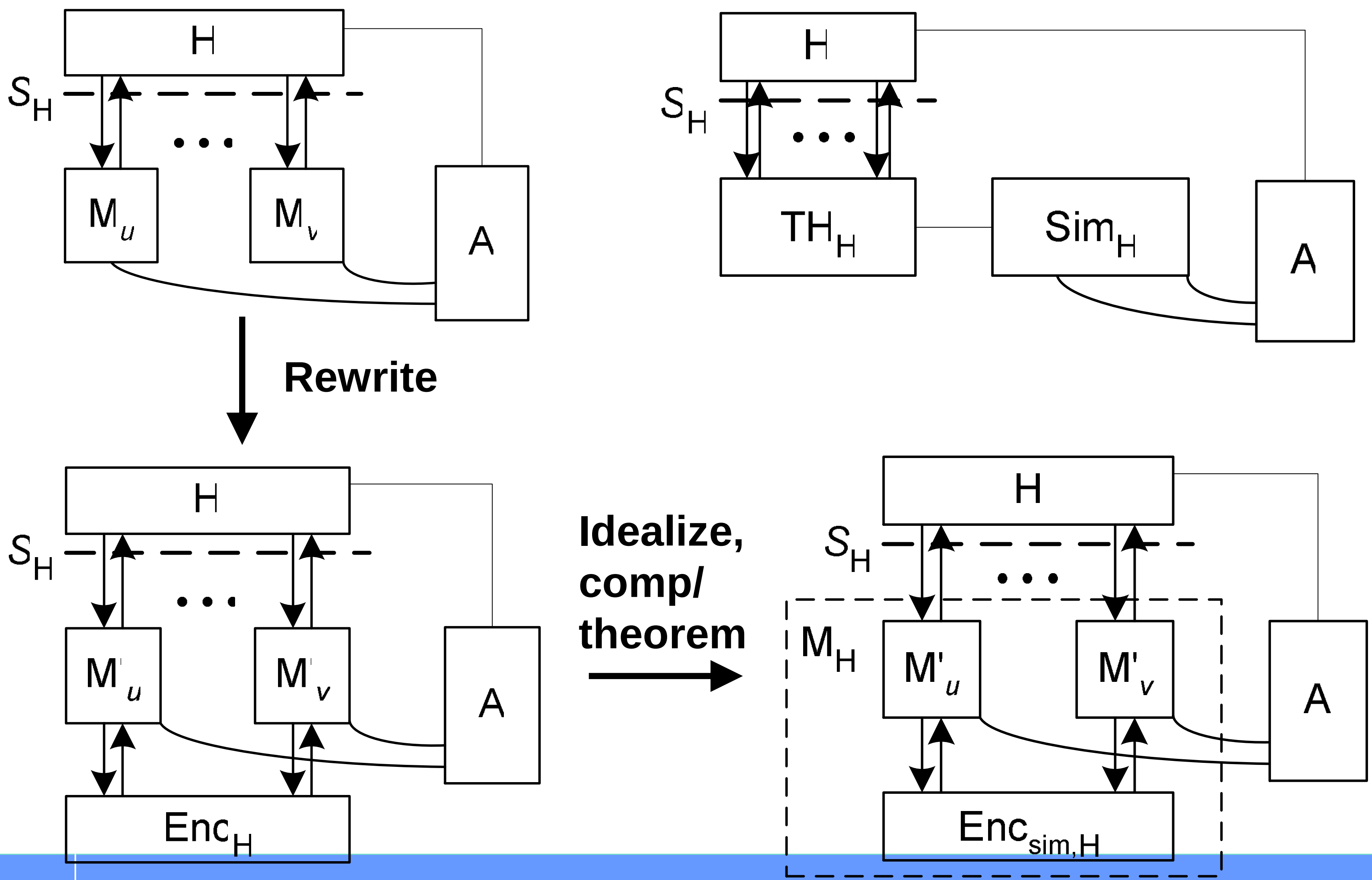
Main Additions to Given Cryptosystems

- **Type tags**
- **Tagging with keys**
- **Additional randomization (e.g., needed when correct machines use A's keys)**

The Simulator

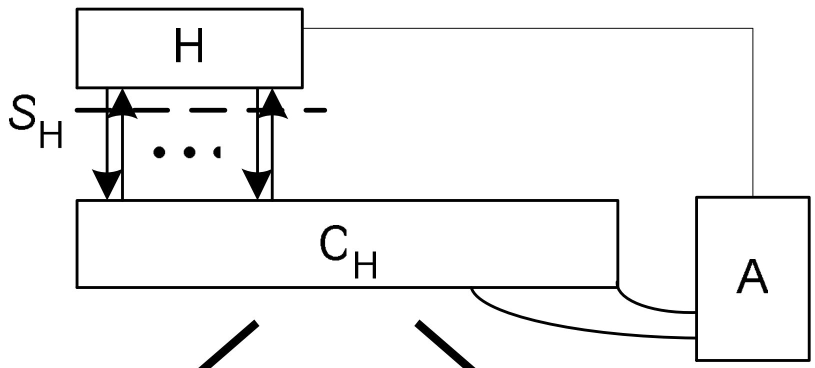


Proof of Correct Simulation (1)



Proof of Correct Simulation (2)

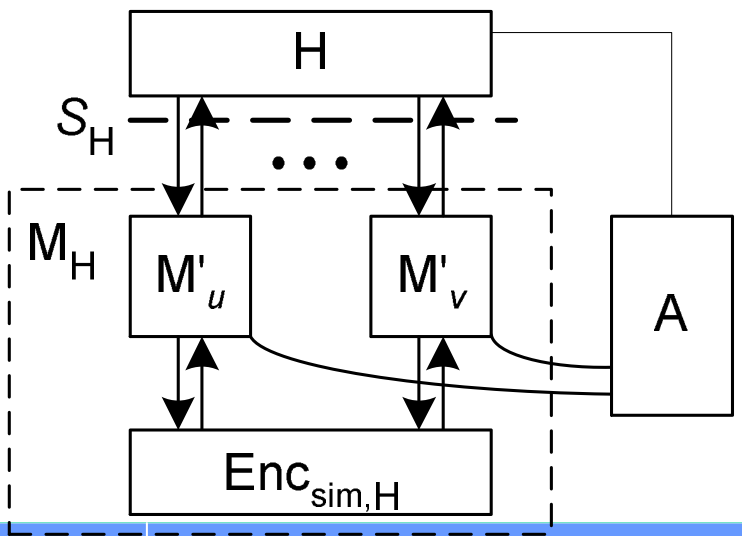
Combined system

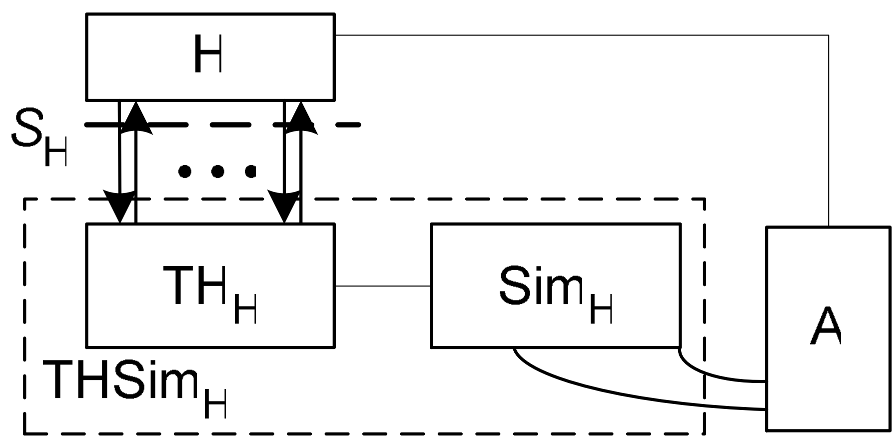


Reduction proofs for collisions, guesses, forgeries

Probabilistic bisimulations

- With error sets (of runs)
- With info-flow analysis





Related Work (until first half of 2005)

	Attacks	Opera-tors	Protocols	Properties	DY version & impl
AR00, AJ01, L01	Passive	1 (pke or ske)	differs	Equivalences	Simple
BPW02, BPW03, BP04	Active	Many	Arbitrary	Simulatability, $\Rightarrow$ Int., non-interf, now nonce, key & payload secrecy	More complex but see L05, BB06
MW04	Active	1 (pke)	Restricted	Integrity	Simple
L04	Active	1 (ske)	Restricted	Equivalences	Simple
CW05	Active	pke, sig	Restricted	Nonce secrecy	Simple
CH05	Active	1 (pke)	Restricted	Key secrecy	Simple

All simple ones come with tool: Specific for “equivalences”, any standard DY tool otherwise

New General Framework for Symbolic Analysis