Cryptographic Logical Relations

— What is the contextual equivalence for cryptographic protocols and how to prove it?

Yu ZHANG

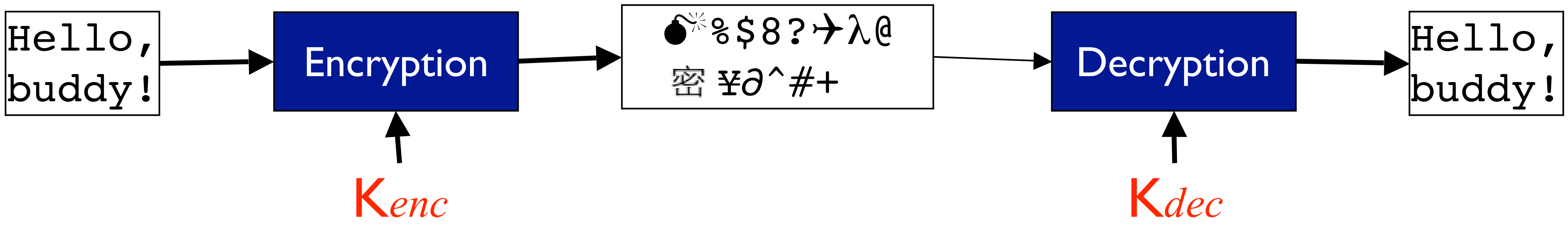
Including joint work with J. Goubault-Larrecq, D. Nowak and S. Lasota

EVEREST, INRIA Sophia-Antipolis

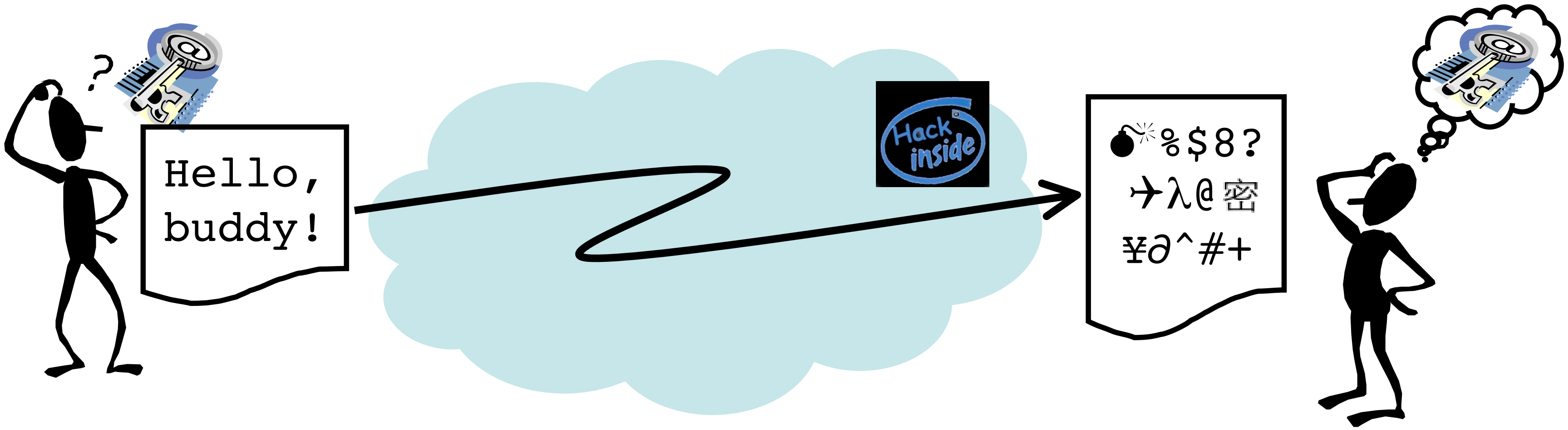
February 12, 2007

Cryptography

Using cryptography to hide information:



But, how to distribute keys on Internet?



The Needham-Schroeder's protocol

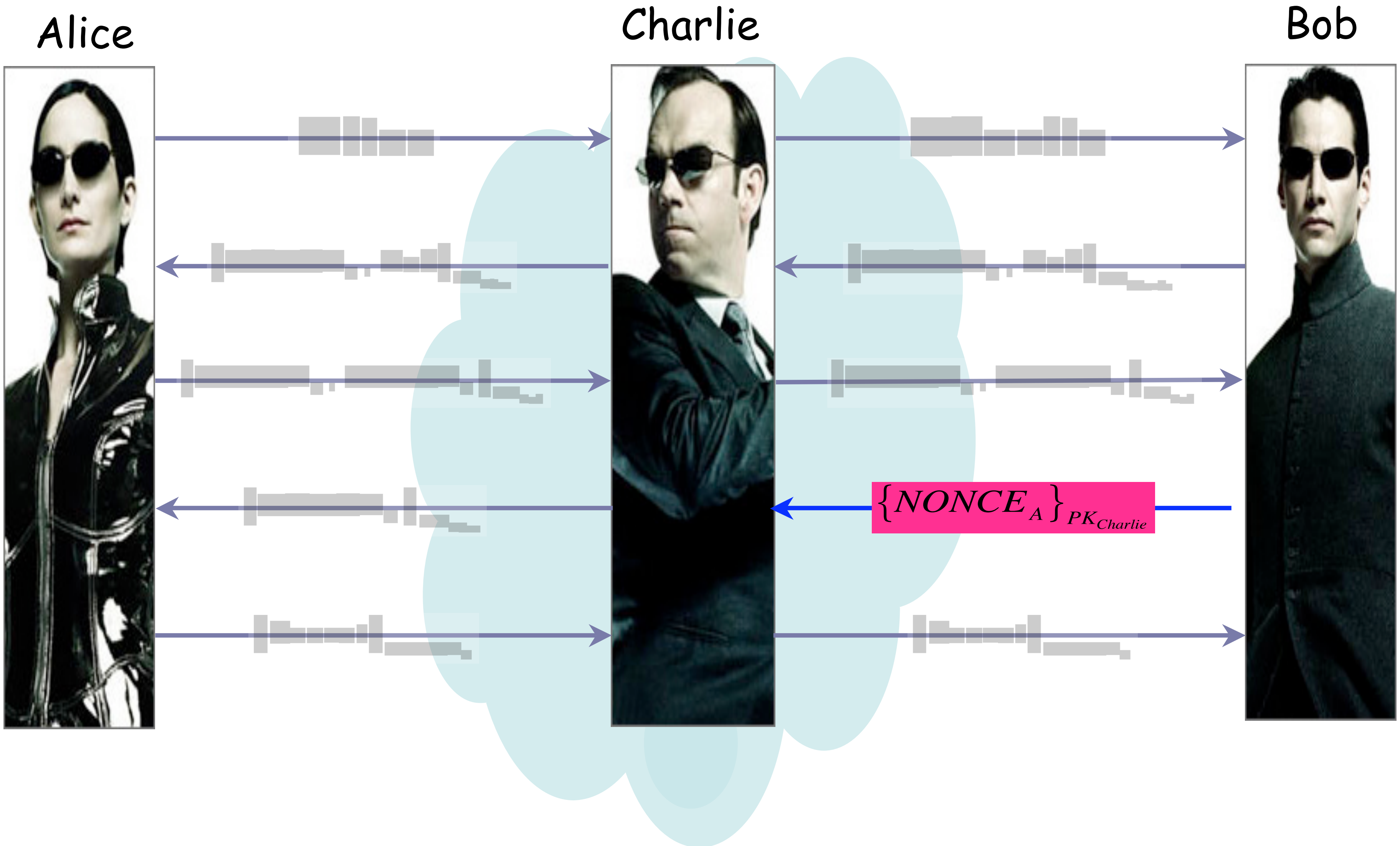
Alice

Bob

$\{Secret\}_{NONCE_A}$

$NONCE_A$ becomes the session key

The Needham-Schroeder's protocol



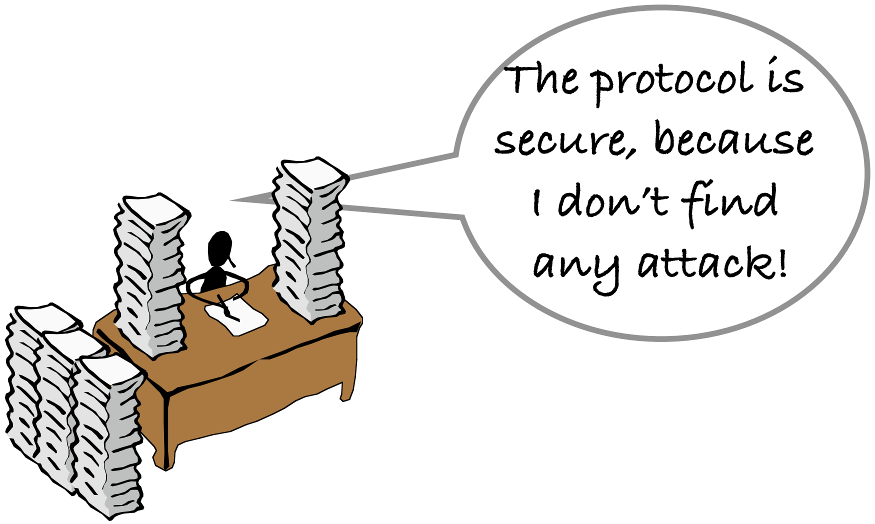
Formal verification

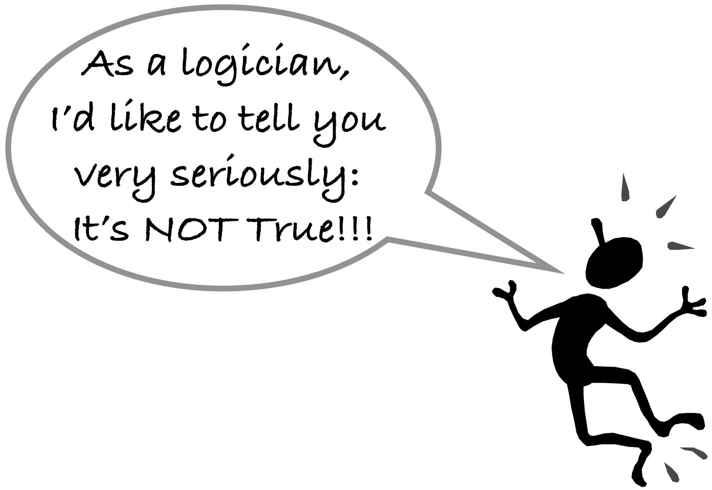
1978 — The invention of the NS protocol [NS 78].

1995 — G. Lowe found the flaw [Lowe 95].

Those who think that their problem can be solved by simply applying cryptography, don't understand cryptography and don't understand their problem.

---- R. Needham





Formal verification

1978 — The invention of the NS protocol [NS 78].

1995 — G. Lowe found the flaw [Lowe 95].

Verify security properties with formal methods.





Formal verification community



SECSI


AVISPA

Secrecy by contextual equivalence

Secrecy: for every messages m_1 and m_2, $Protocol(m_1) \approx Protocol(m_2)$.

Spi-Calculus: with bisimulations [Abadi & Gordon 97].

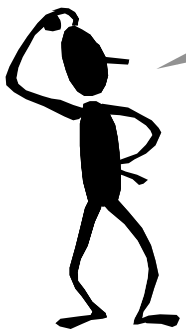
Cryptographic λ-calculus: with logical relations [Sumii & Pierce 02].

Higher-order functions are taken into account.

Motivation

We keep on using the λ-calculus approach.

Sumii and Pierce's logical relations are somehow ad-hoc. Is there a systematic way to construct these logical relations?



And, to what extent can we rely on this method? If logical relations fail in proving the secrecy property, can we say that protocol is NOT secure?

Related work and our contribution

Side-effects	Logical relations
	1980, invention of logical relations [Plotkin 80]
1989, computational λ-calculus [Moggi 89, Moggi 90]	
	1992~93, categorical construction [Ma & Reynolds 92, Mitchell & Scedrov 93]
	1993~94, operational logical relations for name creation [Pitts & Stark 93]
	2002, logical relations for encryption [Sumii & Pierce 02]
2002, logical relations for computational λ-calculus [Goubault-Larrecq, Lasota & Nowak 02]	
	2003, denotational logical relations for key generation [Zhang & Nowak 03]
	2004, lax cryptographic logical relations [Goubault-larrecq, Lasota, Nowak & Zhang 04]
2005, completeness of monadic logical relations [Lasota, Nowak & Zhang 06]	

Outline

- The cryptographic metalanguage
- Denotational semantics
- Cryptographic logical relations
- Contextual equivalence

Cryptographic Logical Relations

- ☑ Introduction
- ☞ The cryptographic metalanguage
- ☐ Denotational semantics
- ☐ Cryptographic logical relations
- ☐ Contextual equivalence
- ☐ Conclusion

Syntax (i) — Types

Based on Moggi's computational λ-calculus — a nice framework for reasoning about side-effects, including key generation.

$$\begin{array}{lcl} \tau & ::= & \cdots \\ & | & \mathsf{key} \\ & | & \mathsf{msg} \\ & | & T\tau \end{array}$$

Type for computations, from Moggi's language

- A computation may generate fresh keys.

Syntax (ii) — Terms

$$
\begin{array}{lll}
t & ::= & \cdots \\
& | & \texttt{new} \\
& | & \texttt{enc}(t, k) \\
& | & \texttt{dec}(t, k) \\
& & \cdots \\
& | & \texttt{val}(t) \\
& | & \texttt{let } x \Leftarrow t_1 \texttt{ in } t_2
\end{array}
$$

generation of fresh key,
from Stark's metalanguage

trivial computation and
sequential computation,
from Moggi's language

Syntax (ii) — Typing rules

$$\frac{}{\Gamma \vdash \mathtt{new} : \mathsf{T}\mathsf{key}}\ (New)$$

$$\frac{\Gamma \vdash t : \tau}{\Gamma \vdash \mathtt{val}(t) : \mathsf{T}\tau}\ (Val) \qquad \frac{\Gamma \vdash t_1 : \mathsf{T}\tau \qquad \Gamma, x : \tau \vdash t_2 : \mathsf{T}\tau'}{\Gamma \vdash \mathtt{let}\ x \Leftarrow t_1\ \mathtt{in}\ t_2 : \mathsf{T}\tau'}\ (Let)$$

$$\frac{\Gamma \vdash t : \mathsf{msg} \qquad \Gamma \vdash k : \mathsf{key}}{\Gamma \vdash \mathtt{enc}(t, k) : \mathsf{msg}}\ (Enc) \qquad \frac{\Gamma \vdash t : \mathsf{msg} \quad \Gamma \vdash k : \mathsf{key}}{\Gamma \vdash \mathtt{dec}(t, k) : \mathsf{opt}[\mathsf{msg}]}\ (Dec)$$

Modeling asymmetric cryptography

Public key cryptography can be modeled using functions [Sumii & Pierce 02]:

- If k is a private key, then the public key is:

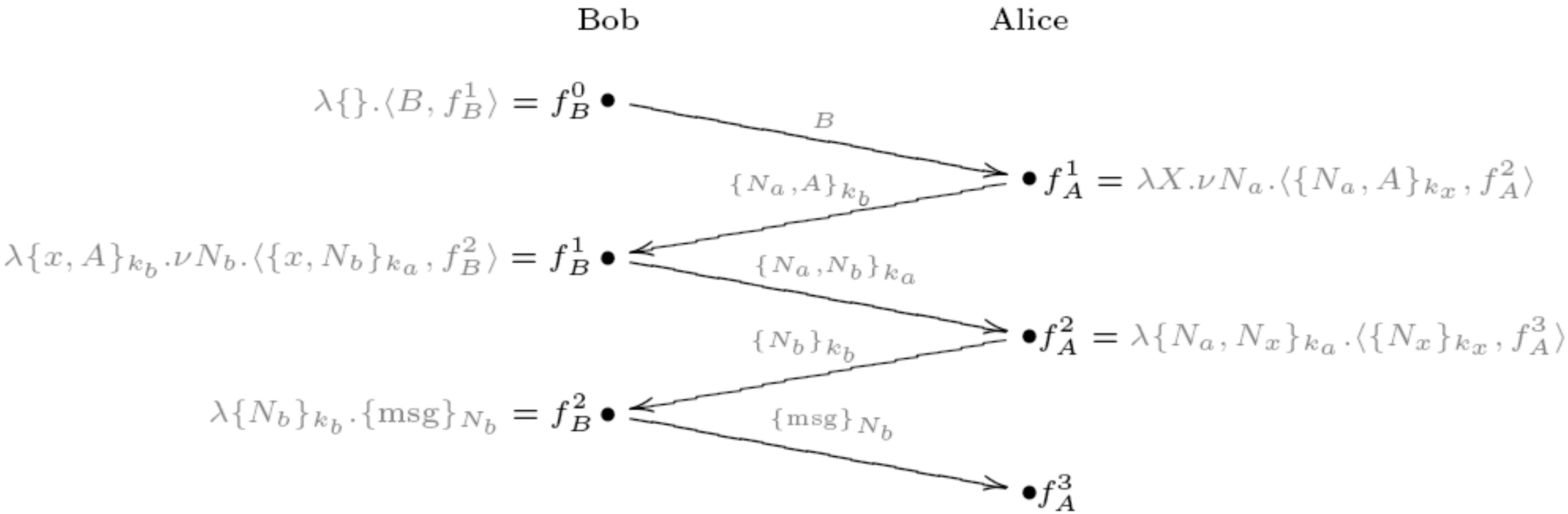
$$PK(k) = \lambda x.\mathtt{enc}(x, k)$$

- Encrypt a message with a public key:

$$Enc(m, PK(k)) = (\lambda x.\mathtt{enc}(x, k))m$$

Encoding of protocols

- Principals as functions.
- Interactions as function applications.



- The protocol is a tuple of functions:

$$P(\mathit{secret}) = \langle f_{Alice}, f_{Bob}, \ldots\rangle$$

- An attack is a function F:

$$F(P(\mathit{secret})) = \mathit{secret}$$

Cryptographic Logical Relations

- ☑ Introduction
- ☑ The cryptographic metalanguage
- ☞ Denotational semantics
- ☐ Cryptographic logical relations
- ☐ Contextual equivalence
- ☐ Conclusion

Modeling cryptography

$[\![\text{key}]\!]$ — a set of keys.

function symbol

An encrypted message is written as $e(v, k)$.

plain-text

key

$$[\![\mathtt{enc}(t_1, t_2)]\!] \quad = \quad e([\![t_1]\!], [\![t_2]\!])$$

$$[\![\mathtt{dec}(t_1, t_2)]\!] \quad = \quad \begin{cases} v, & \text{if } [\![t_1]\!] = e(v, k) \text{ and } [\![t_2]\!] = k \\ \bot, & \text{otherwise.} \end{cases}$$

Computations as monads

- According to Moggi, side-effects can be modeled by monads [Moggi 89].
  - Concrete monads: exceptions, non-determinism, ...
- Fresh key generation is seen as a side-effect.
- Key generation monad: computations might generate fresh keys.
  - Stark uses this monad to interpret his language for name creation [Stark 94].

Stark's model

A functor category $\mathbf{Set}^{\mathcal{I}}$ with a monad $\mathbf{T}$:

- $\mathcal{I}$ — category of finite sets and injections.
  - A set represents a computation stage.
- Denotations are defined over a set of keys.
- Computations are interpreted as

$$\mathbf{T}[\![\tau]\!]s = \{[s', a] \mid s' \in \mathcal{I}, a \in [\![\tau]\!](s + s')\}$$

fresh keys generated during the computation

result of the computation

We use Stark's model to interpret our metalanguage.

Cryptographic Logical Relations

- ☑ Introduction
- ☑ The cryptographic metalanguage
- ☑ Denotational semantics
- ☞ Cryptographic logical relations
- ☐ Contextual equivalence
- ☐ Conclusion

What is a logical relation?

- A logical relation is a family of relations, each indexed by a type.

$$n_1 \; \mathcal{R}_{\mathsf{int}} \; n_2 \quad \Leftrightarrow \quad n_1 = n_2$$

$$\langle a_1, b_1 \rangle \; \mathcal{R}_{\tau \times \tau'} \; \langle a_2, b_2 \rangle \quad \Leftrightarrow \quad a_1 \; \mathcal{R}_{\tau} \; a_2 \;\; \& \;\; b_1 \; \mathcal{R}_{\tau'} \; b_2$$

- Two functions f_1 and f_2 are related iff

$$(\forall a_1, a_2) \; a_1 \; \mathcal{R}_{\tau} \; a_2 \Rightarrow f_1(a_1) \; \mathcal{R}_{\tau'} \; f_2(a_2)$$

- **Basic Lemma**
  - If the denotation of each constant is related to itself, denotations of every term in related environments are related.
  - Basic Lemma helps us to prove contextual equivalence.

What is a cryptographic logical relation?

- The sprit of Sumii and Pierce's logical relations: A cryptographic logical relation

must relate encryption with itself, and relate decryption with itself.

Relations for base types

- Only keys that are accessible to attackers are related [Sumii & Pierce 02, Abadi & Gordon 97]:

$$k_1\ \mathcal{R}^{fr}_{\mathsf{key}}\ k_2 \Leftrightarrow k_1 = k_2\ \&\ k_1, k_2 \in fr$$

$fr \subseteq [\![\mathsf{key}]\!]$– the set of disclosed keys.

- Encrypted messages are then divided into two parts

$$\boxed{\{e(v,k) \mid v \in [\![\mathsf{msg}]\!], k \in fr\}} \cup \boxed{\{e(v,k) \mid v \in [\![\mathsf{msg}]\!], k \notin fr\}}$$

$$e(v_1,k_1)\ \mathcal{R}^{fr,\varphi}_{\mathsf{msg}}\ e(v_2,k_2) \Leftrightarrow \boxed{v_1\ \mathcal{R}^{fr,\varphi}_{\mathsf{msg}}\ v_2\ \ \&\ \ k_1\ \mathcal{R}^{fr,\varphi}_{\mathsf{key}}\ k_2} \text{ or } \boxed{k_1, k_2 \notin fr\ \ \&\ \ (v_1, v_2) \in \varphi(k_1, k_2)}$$

built by induction on message structure

fixed by the parameter φ

φ — parameter of the logical relation, fixing the relation between secret messages [Sumii & Pierce 02].

Logical relations for monadic types

- Categorical construction of logical relation for monadic types [Goubault-Larrecq et al. 02].

  But what is the category for constructing logical relations?

- A logical relation constructed over $\boldsymbol{Set}^{\mathcal{I}}$:

$$\mathcal{R}^s_\tau \quad \subseteq [\![\tau]\!]s \times [\![\tau]\!]s$$

  - Kripke logical relation — logical relations defined over functor categories [Mitchell & Moggi 91].
  - $s \in \mathcal{I}$ s called a "world", representing a computation stage.
  - Two functions are related iff they take related arguments at any larger world to related results.

- Logical relations derived over $\boldsymbol{Set}^{\mathcal{I}}$ are too weak with naïve relations for keys:

$$k_1 \ \mathcal{R}^s_{\mathsf{key}} \ k_2 \Leftrightarrow k_1 = k_2$$

How to represent the parameter fr ?

The “frame” category

Formalize the parameter fr in the category $\mathcal{I}^{\rightarrow}$ [ZN 03]:

- objects are tuples $\langle w, i, s \rangle$;
- morphisms are pairs of injections (j, l) such that the following diagram commutes:

disclosed keys

$$\begin{tikzcd} w \arrow[r, "i"] \arrow[d, "j"'] & s \arrow[d, "l"] \\ w' \arrow[r, "i'"] & s' \end{tikzcd}$$

all keys that have been created

fr Becomes $i(w)$:

$$k_1 \; \mathcal{R}_{\text{key}}^{\langle w,i,s \rangle} \; k_2 \Leftrightarrow k_1 = k_2 \;\&\; k_1, k_2 \in i(w)$$

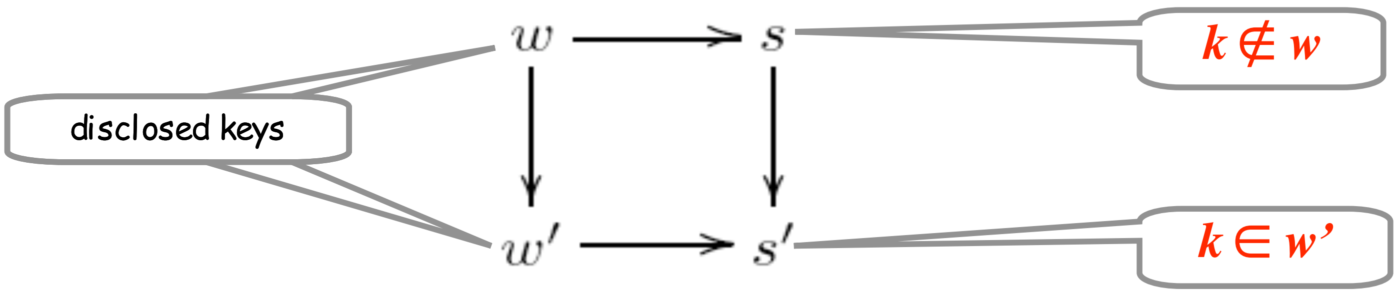
Logical relations over $Set^{\mathcal{I}^{\to}}$

$\mathcal{R}_\tau^{\langle w,i,s\rangle,\varphi} \subseteq [\![\tau]\!]s \times [\![\tau]\!]s$ (using the general construction of [GLLN02]).

- Basic Lemma holds, but only for a very limited set of φ .
- This logical relation fails in relating equivalent programs:

$$\texttt{let } k \Leftarrow \texttt{new in val}(\langle\{\texttt{true}\}_k,\ \{\texttt{false}\}_k,\ \lambda\{x\}_k.x\rangle)$$

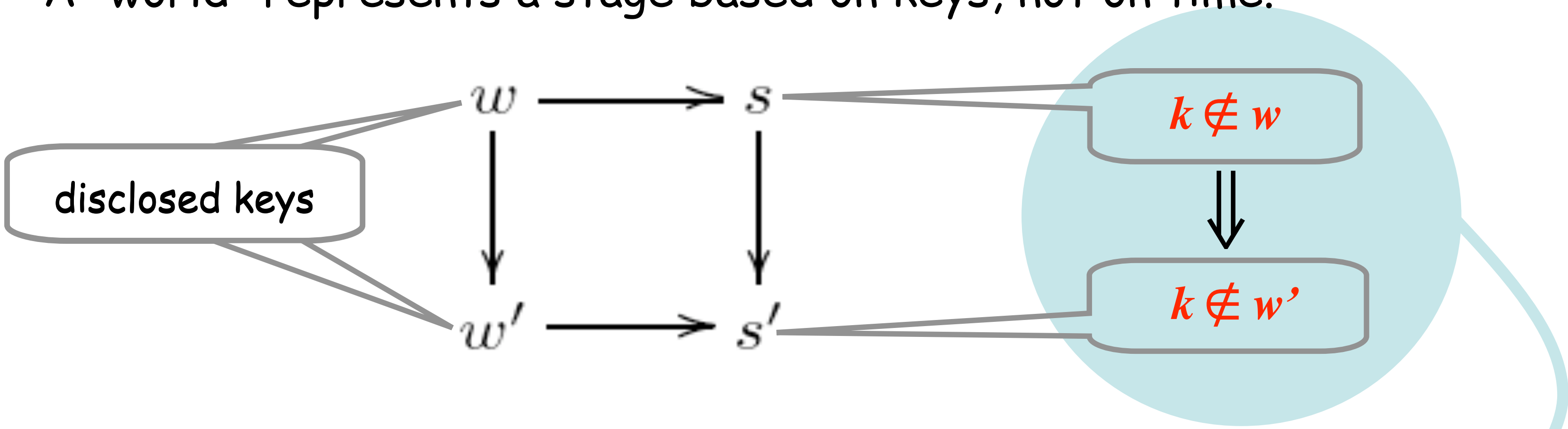
$$\texttt{let } k \Leftarrow \texttt{new in val}(\langle\{\texttt{false}\}_k,\ \{\texttt{true}\}_k,\ \lambda\{x\}_k.\texttt{not}(x)\rangle)$$



Secret keys get known by attackers at a larger "world".

The “frame” category (revised)

- In our model, secret keys must NOT be exposed at any larger “world”.
  - A “world” represents a stage based on keys, not on time.



- Category $\mathcal{PI}^{\rightarrow}$:

  the subcategory of $\mathcal{I}^{\rightarrow}$ where every

$$\begin{tikzcd} w \arrow[r, "i"] \arrow[d, "j"'] & s \arrow[d, "l"] \\ w' \arrow[r, "i'"] & s' \end{tikzcd}$$

is a pull-back.

Cryptographic logical relations

- Cryptographic logical relations derived over $\boldsymbol{Set}^{\mathcal{PI}^{\rightarrow}}$:

$$\mathcal{R}_{\tau}^{\langle w,i,s\rangle,\varphi} \subseteq [\![\tau]\!]s \times [\![\tau]\!]s$$

  - Cipher function φ — a group of "world"-indexed functions, each determining the relation between secret cipher-texts at the "world".
  - Basic Lemma holds for a non-trivial set of cipher functions.
  - Recognize Pitts and Stark's operational logical relations for name creation.

Cryptographic Logical Relations

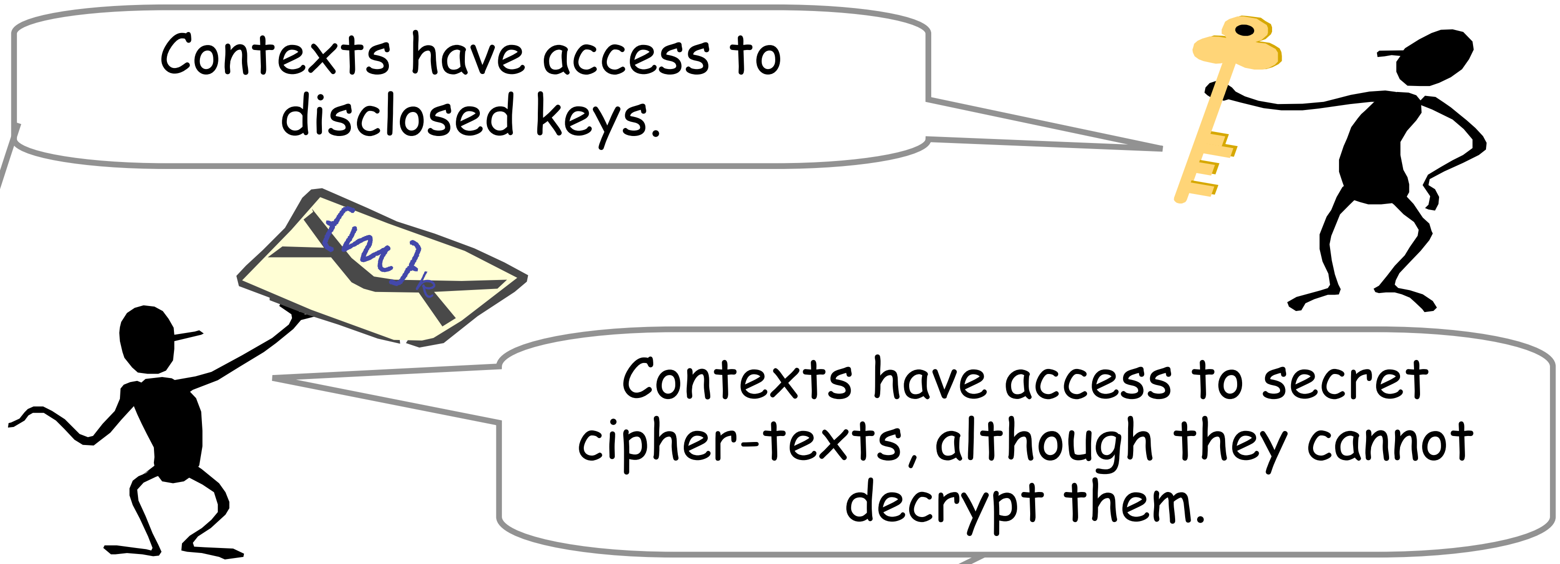
- ☑ Introduction
- ☑ The cryptographic metalanguage
- ☑ Denotational semantics
- ☑ Cryptographic logical relations
- ☞ Contextual equivalence
- ☐ Conclusion

Contexts for cryptographic protocols

- In the computational lambda-calculus, contexts are allowed to do computations:

$$x : \tau \vdash \mathcal{C} : \mathsf{T}o$$

- Contexts represent exactly the knowledge of attackers:



$$\overline{w : \mathsf{key}}, \overline{m : \mathsf{msg}}, x : \tau \vdash \mathcal{C} : \mathsf{T}o$$

Cryptographic contextual equivalence

$$\approx_{\tau}^{\langle w,i,s\rangle,\kappa} \subseteq [\![\tau]\!]s \times [\![\tau]\!]s$$

defined using category $\mathcal{PI}^{\rightarrow}$:

- $\overline{w' : \mathsf{key}}, \overline{m : \mathsf{msg}}, x : \tau \vdash \mathcal{C} : \mathsf{T}o$ holds;
- κ — context knowledge, sets of secret cipher-texts that contexts can access;
- κ-honest environment, mapping every message variable to a cipher-text in κ.

Verifying the secrecy property

- **Secrecy property:**

$$\forall\ msg_1,\ msg_2,\quad Protocol(msg_1) \approx Protocol(msg_2)$$

- **Theorem:**

  Cryptographic logical relations are sound:

$$\mathcal{R}_{\mathcal{T}}^{\langle w,i,s\rangle} \Rightarrow \approx_{\mathcal{T}}^{\langle w,i,s\rangle}$$

- **Proposition:**

  This technique shows that Lowe's fixed version of the Needham-Shroeder protocol satisfies the secrecy property (for multi-sessions) .

Completeness

- A logical relation $\mathcal{R}_\tau$ is complete if $\approx_\tau \;\Rightarrow\; \mathcal{R}_\tau$
- Completeness for monadic logical relations is hard to achieve, even for first-order types.

Our results:

- The cryptographic logical relations are complete for types:

$$\tau_c^1 \quad ::= \quad \mathsf{key} \mid \mathsf{msg} \mid \mathsf{Tkey} \mid \mathsf{key} \rightarrow \tau_c^1 \mid \mathsf{msg} \rightarrow \tau_c^1$$

- A lax logical relation that is complete for all types.

Decidability

- In general, contextual equivalence in the cryptographic metalanguage is undecidable.
- Cryptographic logical relations are decidable for types:

$$\tau^1_d ::= \mathsf{key} \mid \mathsf{msg} \mid \mathsf{T}\tau^1_d \mid \mathsf{key} \to \tau^1_d$$

- Contextual equivalence is decidable for types:

$$\tau^1_\approx ::= \mathsf{key} \mid \mathsf{msg} \mid \mathsf{Tkey} \mid \mathsf{key} \to \tau^1_\approx$$

Cryptographic Logical Relations

- ☑ Introduction
- ☑ The cryptographic metalanguage
- ☑ Denotational semantics
- ☑ Cryptographic logical relations
- ☑ Contextual equivalence
- ☞ Conclusion

Main results

- The category $\boldsymbol{Set}^{\mathcal{PI}^{\rightarrow}}$ for deriving cryptographic logical relations.
- A proper notion of contextual equivalence for cryptographic protocols.
- Cryptographic logical relations:
  - sound (can deduce contextual equivalence);
  - complete for types:

$$\tau_c^1 \quad ::= \quad \mathsf{key} \mid \mathsf{msg} \mid \mathsf{Tkey} \mid \mathsf{key} \to \tau_c^1 \mid \mathsf{msg} \to \tau_c^1$$

- A complete lax logical relation.
- Decidability for contextual equivalence for types:

$$\tau_{\approx}^1 ::= \mathsf{key} \mid \mathsf{msg} \mid \mathsf{Tkey} \mid \mathsf{key} \to \tau_{\approx}^1$$

Future work

- On programming languages:
  - Extend the model for dealing with recursion.
  - Freshness: nominal techniques based on FM-sets (name-swapping) [Pitts et al.].
- On security:
  - Protocols aiming at other security properties, e.g., anonymity.
  - The computational model:
    - Lambda-calculi might be a better language for expressing games, oracle calls, etc.
    - Typing systems enforcing complexity constraints [Hofmann 1997, Mitchell et al. 1998]
    - Logical relations might help!