

Computational Soundness of Symbolic Security Proofs

Steve Kremer

LSV, ENS Cachan & CNRS & INRIA Saclay — Île de France

CosyProofs 2010

Advertisement

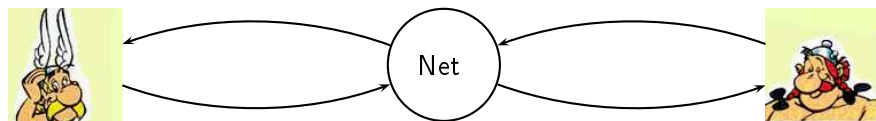
Post-doc position (18 months) at LORIA, Nancy with **Véronique Cortier**

Project **AVOTÉ**: formal analysis of electronic voting protocols

More info at:

<http://www.loria.fr/~cortier/postdoc-AVOTE.pdf>

Cryptographic protocols



Protocols

↔ rules describing the exchange of messages

Goal

↔ secure communications: *secret, authentication, anonymity* ...

Applications

↔ mobile phones, electronic voting, homebanking, electronic commerce ...

Cryptographic protocols



Protocols

↔ rules describing the exchange of messages

Goal

↔ secure communications: *secret, authentication, anonymity* ...

Applications

↔ mobile phones, electronic voting, homebanking, electronic commerce ...

Two approaches for analyzing cryptographic protocols

Symbolic, “Dolev-Yao” approach



- ▶ data are represented as **terms**
- ▶ **idealized** adversary and cryptography represented as deduction rules or equational theories,
- ▶ **automated tools** for analyzing large, complex protocols with multiple or unbounded number of sessions

Computational approach



- ▶ data are represented as **bitstrings**
- ▶ adversaries are **PPT Turing Machines**
- ▶ cryptographic primitives are **PT algorithms**; adversary has **negligible probability to break security**
- ▶ proofs are **by hand and error-prone**

Two approaches for analyzing cryptographic protocols

Symbolic, “Dolev-Yao” approach



- ▶ data are represented as **terms**
- ▶ **idealized** adversary and cryptography represented as deduction rules or equational theories,
- ▶ **automated tools** for analyzing large, complex protocols with multiple or unbounded number of sessions

Computational approach

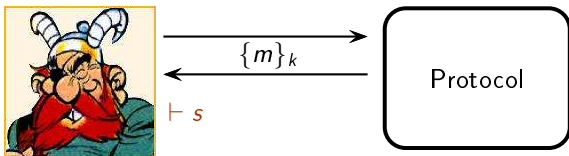


- ▶ data are represented as **bitstrings**
- ▶ adversaries are **PPT Turing Machines**
- ▶ cryptographic primitives are **PT algorithms**; adversary has **negligible probability to break security**
- ▶ proofs are **by hand and error-prone**

Goal: combine the **advantages of both approaches**, i.e. automatic proofs of complex protocols with strong guarantees in a cryptographic model

Different modelling of properties: the case of secrecy

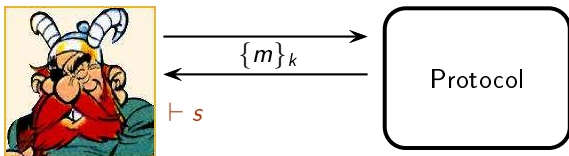
Formal model : property on traces



A data s is secret if the adversary cannot deduce s .

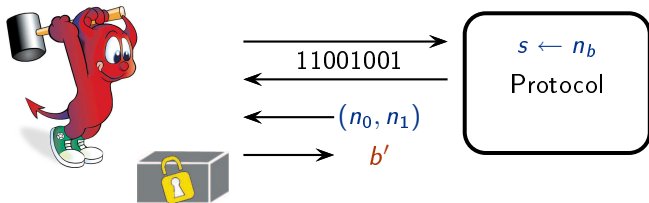
Different modelling of properties: the case of secrecy

Formal model : property on traces



A data s is secret if the adversary cannot deduce s .

Computational model : indistinguishability experiment - generate (n_0, n_1)



The data s is secret if $\Pr[\text{Exp}^1 = 1] - \Pr[\text{Exp}^0 = 1]$ is negligible.

Part I

The Abadi-Rogaway result : formal model

Messages

Messages are modelled by the free algebra over the signature

enc/2, pair/2

and the constants **Bool** and **Keys** where

Bool = $\{0, 1\}$ **Keys** = $\{k, k_1, k_2, \dots, k', k'', \dots\}$

Messages are defined by the following grammar:

$M, N ::=$	termes
K	$K \in \mathbf{Keys}$
$0, 1$	Bool
$\langle M, N \rangle$	pair
$\{M\}_K$	enc ($K \in \mathbf{Keys}$)

Dolev-Yao attacker capabilities: deduction

Message m is deducible from the set of messages $E \supset \{0, 1\}$,
 $E \vdash m$ if:

$$\frac{m \in E}{E \vdash m} \text{ (Ax)}$$

$$\frac{E \vdash \langle m_1, m_2 \rangle}{E \vdash m_1} \text{ (Pr1)}$$

$$\frac{E \vdash \langle m_1, m_2 \rangle}{E \vdash m_2} \text{ (Pr2)}$$

$$\frac{E \vdash m_1 \quad E \vdash m_2}{E \vdash \langle m_1, m_2 \rangle} \text{ (P)}$$

$$\frac{E \vdash m \quad E \vdash k}{E \vdash \{m\}_k} \text{ (Enc)}$$

$$\frac{E \vdash \{m\}_k \quad E \vdash k}{E \vdash m} \text{ (Dec)}$$

Dolev-Yao attacker capabilities: deduction

Message m is deducible from the set of messages $E \supset \{0, 1\}$,
 $E \vdash m$ if:

$$\frac{m \in E}{E \vdash m} \text{ (Ax)}$$

$$\frac{E \vdash \langle m_1, m_2 \rangle}{E \vdash m_1} \text{ (Pr1)}$$

$$\frac{E \vdash \langle m_1, m_2 \rangle}{E \vdash m_2} \text{ (Pr2)}$$

$$\frac{E \vdash m_1 \quad E \vdash m_2}{E \vdash \langle m_1, m_2 \rangle} \text{ (P)}$$

$$\frac{E \vdash m \quad E \vdash k}{E \vdash \{m\}_k} \text{ (Enc)}$$

$$\frac{E \vdash \{m\}_k \quad E \vdash k}{E \vdash m} \text{ (Dec)}$$

Sometimes deduction is not sufficient:

$$t_1 = \{0\}_k$$

$$t_2 = \{1\}_k$$

We expect t_1 and t_2 to be **indistinguishable**.

Patterns

Patterns extend terms by the symbol \square , which intuitively represents encryptions for which the attacker does not know the key.

$P, Q ::=$	patterns
K	$K \in \mathbf{Keys}$
$0, 1$	Bool
$\langle P, Q \rangle$	pair
$\{P\}_K$	enc ($K \in \mathbf{Keys}$)
\square	

Patterns

Patterns extend terms by the symbol \square , which intuitively represents encryptions for which the attacker does not know the key.

$P, Q ::=$	patterns
K	$K \in \text{Keys}$
$0, 1$	Bool
$\langle P, Q \rangle$	pair
$\{P\}_K$	enc ($K \in \text{Keys}$)
\square	

We define the function $p(M, C)$ which takes a term M and a set of keys C as input and outputs a pattern

$p(K, C)$	$= K$	$K \in \text{Keys}$
$p(b, C)$	$= b$	$b \in \text{Bool}$
$p(\langle M, N \rangle, C)$	$= \langle p(M, C), p(N, C) \rangle$	
$p(\{M\}_K, C)$	$= \{p(M, C)\}_K$	si $K \in C$
$p(\{M\}_K, C)$	$= \square$	si $K \notin C$

Patterns (2)

We define the pattern of a term as follows

$$pat(M) = p(M, \underbrace{\{K \in \mathbf{Keys} \mid M \vdash_{\mathcal{I}_{DY}} K\}}_{\text{keys deducible from } M})$$

Example

$$\begin{aligned} pat(\langle \mathbf{0}, \mathbf{1} \rangle) &= \\ pat(\langle \langle \mathbf{0} \rangle_{K_1}, \langle \mathbf{1} \rangle_{K_2}, K_1 \rangle) &= \\ pat(\langle \langle \langle K_1 \rangle_{K_2} \rangle_{K_3}, K_3 \rangle) &= \end{aligned}$$

Patterns (2)

We define the pattern of a term as follows

$$pat(M) = p(M, \underbrace{\{K \in \mathbf{Keys} \mid M \vdash_{\mathcal{I}_{DY}} K\}}_{\text{keys deducible from } M})$$

Example

$$\begin{aligned} pat(\langle \mathbf{0}, \mathbf{1} \rangle) &= \langle \mathbf{0}, \mathbf{1} \rangle \\ pat(\langle \langle \mathbf{0} \rangle_{K_1}, \langle \langle \mathbf{1} \rangle_{K_2}, K_1 \rangle \rangle) &= \\ pat(\langle \langle \langle K_1 \rangle_{K_2} \rangle_{K_3}, K_3 \rangle) &= \end{aligned}$$

Patterns (2)

We define the pattern of a term as follows

$$pat(M) = p(M, \underbrace{\{K \in \mathbf{Keys} \mid M \vdash_{\mathcal{I}_{DY}} K\}}_{\text{keys deducible from } M})$$

Example

$$\begin{aligned} pat(\langle \mathbf{0}, \mathbf{1} \rangle) &= \langle \mathbf{0}, \mathbf{1} \rangle \\ pat(\langle \langle \mathbf{0} \rangle_{K_1}, \langle \langle \mathbf{1} \rangle_{K_2}, K_1 \rangle \rangle) &= \langle \langle \mathbf{0} \rangle_{K_1}, \langle \square, K_1 \rangle \rangle \\ pat(\langle \langle \langle \langle K_1 \rangle_{K_2} \rangle_{K_3}, K_3 \rangle \rangle) &= \end{aligned}$$

Patterns (2)

We define the pattern of a term as follows

$$pat(M) = p(M, \underbrace{\{K \in \mathbf{Keys} \mid M \vdash_{\mathcal{I}_{DY}} K\}}_{\text{keys deducible from } M})$$

Example

$$\begin{aligned} pat(\langle \mathbf{0}, \mathbf{1} \rangle) &= \langle \mathbf{0}, \mathbf{1} \rangle \\ pat(\langle \langle \mathbf{0} \rangle_{K_1}, \langle \mathbf{1} \rangle_{K_2}, K_1 \rangle) &= \langle \langle \mathbf{0} \rangle_{K_1}, \langle \square, K_1 \rangle \rangle \\ pat(\langle \langle \langle K_1 \rangle_{K_2} \rangle_{K_3}, K_3 \rangle) &= \langle \langle \square \rangle_{K_3}, K_3 \rangle \end{aligned}$$

Equivalence of terms

Definition (Equivalence)

Two terms M and N are said to be **equivalent**, $M \equiv N$, iff $pat(M) = pat(N)$.

Problem:

Intuitively, two (random) keys should be equivalent, but $k_1 \not\equiv k_2$.

Equivalence of terms

Definition (Equivalence)

Two terms M and N are said to be **equivalent**, $M \equiv N$, iff $pat(M) = pat(N)$.

Problem:

Intuitively, two (random) keys should be equivalent, but $k_1 \not\equiv k_2$.

Definition (Equivalence up to renaming)

Two terms M and N are **equivalent up to renaming**, noté $M \cong N$ iff there exists a bijection σ on **Keys** such that $M \equiv N\sigma$.

We have that $k_1 \cong k_2$ and $\langle k_1, k_2 \rangle \not\cong \langle k_3, k_3 \rangle$ (no bijection!)

Part II

The Abadi-Rogaway result : computational model

The computational model

As for the formal setting we concentrate on **passive adversaries**

- ▶ no need to describe the protocol execution

In this model

- ▶ Data are **bit strings**
- ▶ Cryptographic primitives are **polynomial-time algorithms**
- ▶ The adversary is an **arbitrary polynomial-time probabilistic Turing machine**
- ▶ Security is expressed as the adversaries success **probability** :
“The probability that an adversary can break the security is negligible”

Preliminaries

Let $\text{String} = \{0, 1\}^*$ be the set of all finite bitstrings

In String we distinguish the sets

- ▶ **Plaintext** : the set of all bitstrings representing possible **plaintexts** including a particular bitstring **0**
- ▶ **Ciphertext** : the set of all bitstrings representing possible **ciphertexts**
- ▶ **Key** : the set of all bitstrings representing possible **keys**

We also define

- ▶ **Coins** = $\{0, 1\}^\omega$: the set of infinite bitstrings, intuitively the **random tape**
- ▶ **Parameter** = 1^* : the set of finite, unary strings, the **complexity parameter**

Encryption scheme

An **encryption scheme** \mathcal{SE} is a triple of algorithms $\mathcal{K}, \mathcal{E}, \mathcal{D}$

- ▶ key generation algorithm $\mathcal{K} : \text{Parameter} \times \text{Coins} \rightarrow \text{Key}$
- ▶ encryption algorithm $\mathcal{E} : \text{Key} \times \text{String} \times \text{Coins} \rightarrow \text{Ciphertext}$
- ▶ decryption algorithm $\mathcal{D} : \text{Key} \times \text{String} \rightarrow \text{Plaintext}$

We suppose that these algorithms can be computed in **polynomial time** in the length of their input (not considering **Coins**)

We use the notation $\mathcal{E}_k(m, r)$ and $\mathcal{D}_k(m, r)$ for $\mathcal{E}(k, m, r)$ and $\mathcal{D}(k, m, r)$

We sometimes omit **Coins** in \mathcal{E} and \mathcal{K} to denote the induced distribution or its support (the set of bitstrings having non-zero probability)

For all $\eta \in \text{Parameter}$, $k \in \mathcal{K}(\eta)$, $r \in \text{Coins}(\eta)$ we require that

$$\mathcal{D}_k(\mathcal{E}_k(m)) = \begin{cases} m & \text{si } m \in \text{Plaintext} \\ \mathbf{0} & \text{si } m \notin \text{Plaintext} \end{cases}$$

Encryption scheme

An **encryption scheme** \mathcal{SE} is a triple of algorithms $\mathcal{K}, \mathcal{E}, \mathcal{D}$

- ▶ key generation algorithm $\mathcal{K} : \text{Parameter} \times \text{Coins} \rightarrow \text{Key}$
- ▶ encryption algorithm $\mathcal{E} : \text{Key} \times \text{String} \times \text{Coins} \rightarrow \text{Ciphertext}$
- ▶ decryption algorithm $\mathcal{D} : \text{Key} \times \text{String} \rightarrow \text{Plaintext}$

Intuitively, the complexity parameter determines the **key length**

Encryption scheme

An **encryption scheme** \mathcal{SE} is a triple of algorithms $\mathcal{K}, \mathcal{E}, \mathcal{D}$

- ▶ key generation algorithm $\mathcal{K} : \text{Parameter} \times \text{Coins} \rightarrow \text{Key}$
- ▶ encryption algorithm $\mathcal{E} : \text{Key} \times \text{String} \times \text{Coins} \rightarrow \text{Ciphertext}$
- ▶ decryption algorithm $\mathcal{D} : \text{Key} \times \text{String} \rightarrow \text{Plaintext}$

Intuitively, the complexity parameter determines the **key length**

Encryption is **probabilistic**

Computational indistinguishability

Definition (Negligible function)

A function $\epsilon : \mathbb{N} \rightarrow \mathbb{R}$ is negligible if

$$\forall n > 0. \exists N_n. \epsilon(\eta) \leq \eta^{-n} \text{ pour } \eta \geq N_n$$

Let $D = \{D_\eta\}$ be a family of distributions over **String**, one for each $\eta \in \text{Parameter}$.

Definition (Indistinguishability)

Let $D = \{D_\eta\}$ and $D' = \{D'_\eta\}$ be two families of distributions. D and D' are **indistinguishable**, denoted $D \approx D'$, if for any probabilistic polynomial-time Turing machine \mathcal{A} the function

$$\text{Adv}^{\text{IND}}(\eta) = \left| \mathbb{P} \left[x \stackrel{R}{\leftarrow} D_\eta : \mathcal{A}(\eta, x) = 1 \right] - \mathbb{P} \left[x \stackrel{R}{\leftarrow} D'_\eta : \mathcal{A}(\eta, x) = 1 \right] \right|$$

is negligible.

Security of an encryption scheme

Definition (Semantic security, IND-CPA)

Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme and $\eta \in \text{Parameter}$. We define the advantage of an adversary \mathcal{A} as

$$\text{Adv}_{\mathcal{SE}, \eta}(\mathcal{A}) = \mathbb{P} \left[k, k' \stackrel{R}{\leftarrow} \mathcal{K}(\eta) : \mathcal{A}^{\mathcal{E}_k(\cdot), \mathcal{E}_{k'}(\cdot)}(\eta) = 1 \right] - \mathbb{P} \left[k \stackrel{R}{\leftarrow} \mathcal{K}(\eta) : \mathcal{A}^{\mathcal{E}_k(\mathbf{0}), \mathcal{E}_k(\mathbf{0})}(\eta) = 1 \right]$$

\mathcal{SE} is **semantically secure** if for any PPT Turing machine \mathcal{A} , $\text{Adv}_{\mathcal{SE}, \eta}(\mathcal{A})$ is negligible (in η).

- ▶ $\mathcal{A}^{\mathcal{O}}$ denotes a Turing machine \mathcal{A} with access to oracles \mathcal{O} (here, encryption oracles).
- ▶ $\mathcal{E}_k(\cdot)$ is an encryption oracle which takes m on input and outputs $c \stackrel{R}{\leftarrow} \mathcal{E}_k(m)$
- ▶ $\mathcal{E}_k(\mathbf{0})$ is an encryption oracle which takes m on input and outputs $c \stackrel{R}{\leftarrow} \mathcal{E}_k(\mathbf{0})$ (hence it is independent of m)

Part III

The Abadi-Rogaway result: Computational soundness

The aim of the soundness result

Find conditions such that

$$\text{If } M \cong N \text{ then } \llbracket M \rrbracket \approx \llbracket N \rrbracket$$

where $\llbracket M \rrbracket$ is the implementation of the term M .

Implementing formal terms

We define an implementation of formal terms. Let \mathcal{SE} be an encryption scheme and $\eta \in \text{Parameter}$. We associate to M a distribution $\llbracket M \rrbracket_{\mathcal{SE}, \eta}$ and a family of distributions $\llbracket M \rrbracket_{\mathcal{SE}}$.

Initialize $_{\eta}(M)$

for $K \in \text{Keys}(M)$ do $\tau(K) \stackrel{R}{\leftarrow} \mathcal{K}(\eta)$

Convert(M)

if $M = K$ ($K \in \mathbf{Keys}$) then return ($\tau(K)$, “key”)

if $M = b$ ($b \in \mathbf{Bool}$) then return (b , “bool”)

if $M = \langle M_1, M_2 \rangle$ then return (Convert(M_1), Convert(M_2), “pair”)

if $M = \{M_1\}_K$ then

$x \stackrel{R}{\leftarrow} \text{Convert}(M_1)$; $y \stackrel{R}{\leftarrow} \mathcal{E}_{\tau(K)}(x)$; return(y , “ciphertext”)

Implementing formal terms

We define an implementation of formal terms. Let \mathcal{SE} be an encryption scheme and $\eta \in \text{Parameter}$. We associate to M a distribution $\llbracket M \rrbracket_{\mathcal{SE}, \eta}$ and a family of distributions $\llbracket M \rrbracket_{\mathcal{SE}}$.

Initialize $_{\eta}(M)$

for $K \in \text{Keys}(M)$ do $\tau(K) \stackrel{R}{\leftarrow} \mathcal{K}(\eta)$

Convert(M)

if $M = K$ ($K \in \mathbf{Keys}$) then return ($\tau(K)$, “key”)

if $M = b$ ($b \in \mathbf{Bool}$) then return (b , “bool”)

if $M = \langle M_1, M_2 \rangle$ then return (Convert(M_1), Convert(M_2), “pair”)

if $M = \{M_1\}_K$ then

$x \stackrel{R}{\leftarrow} \text{Convert}(M_1)$; $y \stackrel{R}{\leftarrow} \mathcal{E}_{\tau(K)}(x)$; return(y , “ciphertext”)

- ▶ we generate the keys occurring on M (denoted $\text{Keys}(M)$) by calling \mathcal{K} and store them in the array τ

Implementing formal terms

We define an implementation of formal terms. Let \mathcal{SE} be an encryption scheme and $\eta \in \text{Parameter}$. We associate to M a distribution $\llbracket M \rrbracket_{\mathcal{SE}, \eta}$ and a family of distributions $\llbracket M \rrbracket_{\mathcal{SE}}$.

Initialize $_{\eta}(M)$

for $K \in \text{Keys}(M)$ do $\tau(K) \xleftarrow{R} \mathcal{K}(\eta)$

Convert(M)

if $M = K$ ($K \in \mathbf{Keys}$) then return $(\tau(K), \text{"key"})$

if $M = b$ ($b \in \mathbf{Bool}$) then return $(b, \text{"bool"})$

if $M = \langle M_1, M_2 \rangle$ then return $(\text{Convert}(M_1), \text{Convert}(M_2), \text{"pair"})$

if $M = \{M_1\}_K$ then

$x \xleftarrow{R} \text{Convert}(M_1)$; $y \xleftarrow{R} \mathcal{E}_{\tau(K)}(x)$; return $(y, \text{"ciphertext"})$

- ▶ we generate the keys occurring on M (denoted $\text{Keys}(M)$) by calling \mathcal{K} and store them in the array τ
- ▶ we suppose there exists an implementation of terms **0** and **1**

Implementing formal terms

We define an implementation of formal terms. Let \mathcal{SE} be an encryption scheme and $\eta \in \text{Parameter}$. We associate to M a distribution $\llbracket M \rrbracket_{\mathcal{SE}, \eta}$ and a family of distributions $\llbracket M \rrbracket_{\mathcal{SE}}$.

Initialize $_{\eta}(M)$

for $K \in \text{Keys}(M)$ do $\tau(K) \stackrel{R}{\leftarrow} \mathcal{K}(\eta)$

Convert(M)

if $M = K$ ($K \in \mathbf{Keys}$) then return $(\tau(K), \text{"key"})$

if $M = b$ ($b \in \mathbf{Bool}$) then return $(b, \text{"bool"})$

if $M = \langle M_1, M_2 \rangle$ then return $(\text{Convert}(M_1), \text{Convert}(M_2), \text{"pair"})$

if $M = \{M_1\}_K$ then

$x \stackrel{R}{\leftarrow} \text{Convert}(M_1)$; $y \stackrel{R}{\leftarrow} \mathcal{E}_{\tau(K)}(x)$; return $(y, \text{"ciphertext"})$

- ▶ we generate the keys occurring on M (denoted $\text{Keys}(M)$) by calling \mathcal{K} and store them in the array τ
- ▶ we suppose there exists an implementation of terms **0** and **1**
- ▶ to avoid ambiguities we use tags

Forbidding key cycles

To obtain a soundness result we need to forbid **key cycles**.

Definition (Key Cycle)

Let $K, K' \in \mathbf{Keys}$. K encrypts K' in a term M written $K \succ_M K'$ if $\{N\}_K \in st(M)$ and $K' \in st(N)$.

A term M is **acyclic** iff the relation \succ_M is acyclic.

Example

Forbidding key cycles

To obtain a soundness result we need to forbid **key cycles**.

Definition (Key Cycle)

Let $K, K' \in \mathbf{Keys}$. K encrypts K' in a term M written $K \succ_M K'$ if $\{N\}_K \in st(M)$ and $K' \in st(N)$.

A term M is **acyclic** iff the relation \succ_M is acyclic.

Example

- ▶ Let $M = \langle \{\{\{K_1\}_{K_2}\}_{K_3}, \mathbf{0}\} \rangle$. \succ_M is defined as $K_3 \succ_M K_2 \succ_M K_1$. Hence, M is acyclic.

Forbidding key cycles

To obtain a soundness result we need to forbid **key cycles**.

Definition (Key Cycle)

Let $K, K' \in \mathbf{Keys}$. K encrypts K' in a term M written $K \succ_M K'$ if $\{N\}_K \in st(M)$ and $K' \in st(N)$.

A term M is **acyclic** iff the relation \succ_M is acyclic.

Example

- ▶ Let $M = \langle \{\{\{K_1\}_{K_2}\}_{K_3}, \mathbf{0}\} \rangle$. \succ_M is defined as $K_3 \succ_M K_2 \succ_M K_1$. Hence, M is acyclic.
- ▶ Let $M = \{K\}_K$. We have $K \succ_M K$. M contains a key cycle of size 1.

Forbidding key cycles

To obtain a soundness result we need to forbid **key cycles**.

Definition (Key Cycle)

Let $K, K' \in \mathbf{Keys}$. K encrypts K' in a term M written $K \succ_M K'$ if $\{N\}_K \in st(M)$ and $K' \in st(N)$.

A term M is **acyclic** iff the relation \succ_M is acyclic.

Example

- ▶ Let $M = \langle \{\{K_1\}_{K_2}\}_{K_3}, \mathbf{0} \rangle$. \succ_M is defined as $K_3 \succ_M K_2 \succ_M K_1$. Hence, M is acyclic.
- ▶ Let $M = \{K\}_K$. We have $K \succ_M K$. M contains a key cycle of size 1.
- ▶ Let $M = \langle \{K_1\}_{K_2}, \{K_2\}_{K_1} \rangle$. We have that $K_1 \succ_M K_2$ and $K_2 \succ_M K_1$. M contains a cycle of size 2.

The problem of key cycles

There exist semantically secure schemes that **reveal the encryption key** if the adversary is given a message containing a key cycle.

Suppose that $\mathcal{SE} = (\mathcal{KG}, \mathcal{E}, \mathcal{D})$ is a semantically secure encryption scheme and let $\mathcal{SE}' = (\mathcal{KG}', \mathcal{E}', \mathcal{D}')$ be defined as follows:

$$\mathcal{KG}' = \mathcal{KG}$$

$$\mathcal{E}'_k(m, r) = \begin{cases} \mathcal{E}_k(m, r) & \text{if } m \neq k \\ \text{const} \cdot k & \text{if } m = k \end{cases}$$

$$\mathcal{D}'_k(c) = \begin{cases} \mathcal{D}_k(c) & \text{if } c \neq \text{const} \cdot k \\ k & \text{if } c = \text{const} \cdot k \end{cases}$$

\mathcal{SE}' is semantically secure, but

$$\llbracket \{K\}_K \rrbracket \not\approx \llbracket \{K'\}_K \rrbracket$$

Formal equivalence implies computational soundness

Theorem (Computational soundness of formal equivalence)

Let M and N be two acyclic terms and \mathcal{SE} a semantically secure encryption scheme. If $M \cong N$ then $\llbracket M \rrbracket \approx \llbracket N \rrbracket$.

This theorem allows us to use formal, automated techniques and obtain security guarantees in the computational setting.

Proof: renaming

The first step consists in **renaming keys**.

Let $Keys(M)$ be the set of keys in M .

$$\begin{aligned} recoverable(M) &= \{K \in Keys(M) \mid M \vdash K\} \\ hidden(M) &= Keys(M) \setminus recoverable(M) \end{aligned}$$

Let $\mu = |recoverable(M)|$ and $m = |hidden(M)|$.

Rename keys in $recoverable(M)$ to J_1, \dots, J_μ .

By acyclicity of M we can rename keys in $hidden(M)$ to K_1, \dots, K_m such that $K_i \succ_M K_j$ implies $i > j$.

A “deeper” key gets a smaller index.

Proof : renaming

Example

Let M be the following term (we omit pairs for readability)

$$\{0\}_{K_6} \{K_1 1\}_{K_4} K_2 \{0\}_{K_3} \{K_6\}_{K_4} \{K_1 K_3\}_{K_4} \{111\}_{K_5} 0 \{K_1\}_{K_6} \{K_5\}_{K_2}$$

$Keys(M) =$
 $recoverable(M) =$
 $hidden(M) =$

Proof : renaming

Example

Let M be the following term (we omit pairs for readability)

$$\{0\}_{K_6} \{K_1 1\}_{K_4} K_2 \{0\}_{K_3} \{K_6\}_{K_4} \{K_1 K_3\}_{K_4} \{111\}_{K_5} 0 \{K_1\}_{K_6} \{K_5\}_{K_2}$$

$$\begin{aligned} \text{Keys}(M) &= \{K_1, K_2, K_3, K_4, K_5, K_6\} \\ \text{recoverable}(M) &= \{K_2, K_5\} \\ \text{hidden}(M) &= \{K_1, K_3, K_4, K_6\} \end{aligned}$$

The relation \succ_M (restricted to $\text{hidden}(M)$) is defined by

Proof : renaming

Example

Let M be the following term (we omit pairs for readability)

$$\{0\}_{K_6} \{K_1 1\}_{K_4} K_2 \{0\}_{K_3} \{K_6\}_{K_4} \{K_1 K_3\}_{K_4} \{111\}_{K_5} 0 \{K_1\}_{K_6} \{K_5\}_{K_2}$$

$$\begin{aligned} \text{Keys}(M) &= \{K_1, K_2, K_3, K_4, K_5, K_6\} \\ \text{recoverable}(M) &= \{K_2, K_5\} \\ \text{hidden}(M) &= \{K_1, K_3, K_4, K_6\} \end{aligned}$$

The relation \succ_M (restricted to $\text{hidden}(M)$) is defined by

$$K_4 \succ_M K_1 \quad K_4 \succ_M K_3 \quad K_4 \succ_M K_6 \quad K_6 \succ_M K_1$$

Proof : renaming

Example

Let M be the following term (we omit pairs for readability)

$$\{0\}_{K_6} \{K_1 1\}_{K_4} K_2 \{0\}_{K_3} \{K_6\}_{K_4} \{K_1 K_3\}_{K_4} \{111\}_{K_5} 0 \{K_1\}_{K_6} \{K_5\}_{K_2}$$

$$\begin{aligned} \text{Keys}(M) &= \{K_1, K_2, K_3, K_4, K_5, K_6\} \\ \text{recoverable}(M) &= \{K_2, K_5\} \\ \text{hidden}(M) &= \{K_1, K_3, K_4, K_6\} \end{aligned}$$

The relation \succ_M (restricted to $\text{hidden}(M)$) is defined by

$$K_4 \succ_M K_1 \quad K_4 \succ_M K_3 \quad K_4 \succ_M K_6 \quad K_6 \succ_M K_1$$

Rename

$\{K_1 \rightarrow K_1, K_2 \rightarrow J_1, K_3 \rightarrow K_2, K_4 \rightarrow K_4, K_5 \rightarrow J_2, K_6 \rightarrow K_3\}$ and

get $M' =$

$$\{0\}_{K_3} \{K_1 1\}_{K_4} J_1 \{0\}_{K_2} \{K_3\}_{K_4} \{K_1 K_2\}_{K_4} \{111\}_{J_2} 0 \{K_1\}_{K_3} \{J_2\}_{J_1}$$

Proof: renaming

As $M \cong N$ we have $pat(M) = pat(N\sigma)$. Hence $recoverable(M) = recoverable(N\sigma)$.

As N is acyclic rename keys from $hidden(N)$ to $\{K_1, \dots, K_n\}$ s.t. $n = |hidden(N)|$.

We construct a renaming σ' such that $N' = N\sigma'$, $M' \equiv N'$, $recoverable(M') = recoverable(N') = \{J_1, \dots, J_\mu\}$, $hidden(N') = \{K_1, \dots, K_n\}$ and $K_i \succ_N K_j$ implies $i > j$.

Example

Let N be

$\{11\}_{K_2} \{K_3\}_{K_2} K_1 \{K_3\}_{K_2} \{K_8\}_{K_2} \{1\}_{K_5} \{111\}_{K_3} 0 \{00\}_{K_8} \{K_3\}_{K_1}$

We have that

$recoverable(N) = \{K_1, K_3\}$, $hidden(N) = \{K_2, K_5, K_8\}$. Rename $\{K_1 \rightarrow J_1, K_2 \rightarrow K_3, K_3 \rightarrow J_2, K_5 \rightarrow K_1, K_8 \rightarrow K_2\}$ and get N'

$\{11\}_{K_3} \{J_2\}_{K_3} J_1 \{J_2\}_{K_3} \{K_2\}_{K_3} \{1\}_{K_1} \{111\}_{J_2} 0 \{00\}_{K_2} \{J_2\}_{J_1}$

Proof: hybrid patterns

In the second phase of the proof introduce patterns to form a chain

$$M' = M_m, \dots, M_1, M_0 = N_0, N_1, \dots, N_n = N'$$

where

$$M_i = p(M', \text{recoverable}(M') \cup \{K_1, \dots, K_i\}) \quad 0 \leq i \leq m$$

and

$$N_i = p(N', \text{recoverable}(N') \cup \{K_1, \dots, K_i\}) \quad 0 \leq i \leq n$$

Intuitively M_i respectively N_i is the adversary's view if he would know the hidden keys $\{K_1, \dots, K_i\}$

We indeed have $M_0 = \text{pattern}(M') = \text{pattern}(N') = N_0$ as $M' \equiv N'$.

Proof: hybrid patterns (example)

Example

M'

=

M_4	: $\{0\}_{K_3}$	$\{K_1 1\}_{K_4}$	J_1	$\{0\}_{K_2}$	$\{K_3\}_{K_4}$	$\{K_1 K_2\}_{K_4}$	$\{111\}_{J_2}$	0	$\{K_1\}_{K_3}$	$\{J_2\}_{J_1}$
M_3	: $\{0\}_{K_3}$	\square	J_1	$\{0\}_{K_2}$	\square	\square	$\{111\}_{J_2}$	0	$\{K_1\}_{K_3}$	$\{J_2\}_{J_1}$
M_2	: \square	\square	J_1	$\{0\}_{K_2}$	\square	\square	$\{111\}_{J_2}$	0	\square	$\{J_2\}_{J_1}$
M_1	: \square	\square	J_1	\square	\square	\square	$\{111\}_{J_2}$	0	\square	$\{J_2\}_{J_1}$
M_0	: \square	\square	J_1	\square	\square	\square	$\{111\}_{J_2}$	0	\square	$\{J_2\}_{J_1}$

=

N_0	: \square	\square	J_1	\square	\square	\square	$\{111\}_{J_2}$	0	\square	$\{J_2\}_{J_1}$
N_1	: \square	\square	J_1	\square	\square	$\{1\}_{K_1}$	$\{111\}_{J_2}$	0	\square	$\{J_2\}_{J_1}$
N_2	: \square	\square	J_1	\square	\square	$\{1\}_{K_1}$	$\{111\}_{J_2}$	0	$\{00\}_{K_2}$	$\{J_2\}_{J_1}$
N_3	: $\{11\}_{K_3}$	$\{J_2\}_{K_3}$	J_1	$\{J_2\}_{K_3}$	$\{K_2\}_{K_3}$	$\{1\}_{K_1}$	$\{111\}_{J_2}$	0	$\{00\}_{K_2}$	$\{J_2\}_{J_1}$

=

N'

Proof: associate distributions to patterns

As for terms we associate (families of) distributions to patterns

Idea : the symbol \square is implemented by the encryption of $\mathbf{0}$ with a fresh key.

Extend $\text{Initialize}_\eta(M)$ by

$$\tau(K_0) \stackrel{R}{\leftarrow} \mathcal{K}(\eta)$$

and $\text{Convert}(M)$ by

if $M = \square$ then
 $y \stackrel{R}{\leftarrow} \mathcal{E}_{\tau(K_0)}(\mathbf{0})$
 return(y , "ciphertext")

Proof: by contradiction

We have that

$$\llbracket M \rrbracket_{S\mathcal{E}} = \llbracket M' \rrbracket_{S\mathcal{E}} \quad \llbracket N \rrbracket_{S\mathcal{E}} = \llbracket N' \rrbracket_{S\mathcal{E}}$$

as M and M' (resp. N and N') only differ by renaming of keys

Our goal is to show $\llbracket M' \rrbracket_{S\mathcal{E}} \approx \llbracket N' \rrbracket_{S\mathcal{E}}$

By contradiction, suppose that there exists PPT adversary \mathcal{A} able to distinguish $\llbracket M' \rrbracket_{S\mathcal{E}}$ and $\llbracket N' \rrbracket_{S\mathcal{E}}$

$$\lambda(\eta) = \Pr[y \stackrel{R}{\leftarrow} \llbracket M' \rrbracket_{S\mathcal{E}} : \mathcal{A}(\eta, y) = 1] - \Pr[y \stackrel{R}{\leftarrow} \llbracket N' \rrbracket_{S\mathcal{E}} : \mathcal{A}(\eta, y) = 1]$$

is not negligible

Proof

We define for $1 \leq i \leq m$ and $1 \leq j \leq n$

$$p_i = Pr[y \stackrel{R}{\leftarrow} \llbracket M_i \rrbracket_{S\mathcal{E}} : \mathcal{A}(\eta, y) = 1]$$

$$q_j = Pr[y \stackrel{R}{\leftarrow} \llbracket N_j \rrbracket_{S\mathcal{E}} : \mathcal{A}(\eta, y) = 1]$$

We have that $\lambda = p_m - q_n$ and $p_0 = q_0$ as $pat(M_0) = pat(N_0)$

$$\begin{aligned} \lambda &= (p_m - p_{m-1}) + (p_{m-1} - p_{m-2}) + \cdots + (p_1 - p_0) \\ &\quad + (q_0 - q_1) + (q_1 - q_2) + \cdots + (q_{n-1} - q_n) \end{aligned}$$

By the triangle inequality there exists i . $1 \leq i \leq m$

$$p_i - p_{i-1} \geq \frac{\lambda}{m+n}$$

or there exists j . $1 \leq j \leq n$

$$q_{j-1} - q_j \geq \frac{\lambda}{m+n}$$

An adversary against the encryption scheme

Given \mathcal{A} that distinguishes $\llbracket M_i \rrbracket_{\mathcal{SE}}$ and $\llbracket M_{i-1} \rrbracket_{\mathcal{SE}}$ we define the adversary \mathcal{A}_0 which aims at breaking semantic security

algorithm $\mathcal{A}_0^{f,g}$ (f, g are either $\mathcal{E}_k(\cdot), \mathcal{E}_{k'}(\cdot)$ or $\mathcal{E}_k(0), \mathcal{E}_{k'}(0)$)
for $K \in \text{Keys}(M')$ do $\tau(K) \xleftarrow{R} \mathcal{K}(\eta)$
 $y \xleftarrow{R} \text{Convert2}(M')$
 $b \xleftarrow{R} \mathcal{A}(\eta, y)$

Convert2(M^*)

if $M^* = K$ ($K \in \mathbf{Keys}$) then return ($\tau(K)$, “key”)

if $M^* = b$ ($b \in \mathbf{Bool}$) then return (b , “bool”)

if $M^* = \langle M_1^*, M_2^* \rangle$ then return ($\text{Convert}(M_1^*), \text{Convert2}(M_2^*)$, “pair”)

if $M^* = \{M_1^*\}_K$ then

if $K \in \{J_1, \dots, J_\mu, K_1, \dots, K_{i-1}\}$ then

$x \xleftarrow{R} \text{Convert2}(M_1^*); y \xleftarrow{R} \mathcal{E}_{\tau(K)}(x);$ return(y , “ciphertext”)

if $K = K_i$ then

$x \xleftarrow{R} \text{Convert2}(M_1^*); y \xleftarrow{R} f(x);$ return(y , “ciphertext”)

if $K \in \{K_{i+1}, \dots, K_m\}$ then

$y \xleftarrow{R} g(0);$ return(y , “ciphertext”)

An adversary against the encryption scheme

By construction we have

$$\begin{aligned} p_i(\eta) &= \Pr[k_i, k_0 \stackrel{\mathcal{K}}{\leftarrow} (\eta) : \mathcal{A}_0^{\mathcal{E}_{k_i}(\cdot), \mathcal{E}_{k_0}(\cdot)}(\eta) = 1] \\ p_{i-1}(\eta) &= \Pr[k_0 \stackrel{\mathcal{K}}{\leftarrow} (\eta) : \mathcal{A}_0^{\mathcal{E}_{k_0}(\mathbf{0}), \mathcal{E}_{k_0}(\mathbf{0})}(\eta) = 1] \end{aligned}$$

Hence we have that

$$\begin{aligned} Adv_{\mathcal{SE}, \eta} &= p_i(\eta) - p_{i-1}(\eta) \\ &\geq \lambda(\eta)/(m+n) \\ &> \eta^{-c}/(m+n) \\ &> \eta^{-(c+1)} \quad \forall \eta > (m+n) \end{aligned}$$

$Adv_{\mathcal{SE}, \eta}$ is not a negligible function contradicting semantic security of \mathcal{SE} and concluding the proof.

Part IV

A framework for equational theories

Motivations

An attack on a recursive authentication protocol:
A cautionary tale

[SchneiderRyan'98]

Paradox: an attack on a protocol that was proven correct (no error in the proof!)

- ▶ a more abstract protocol is **proven correct**
- ▶ the actual protocol **implements encryption using exclusive or (\oplus)**
- ▶ **algebraic properties** of \oplus are ignored

Algebraic properties

Recently **many** papers address **algebraic properties**:

[H. Comon-Lundh, R. Treinen: *Easy Intruder Deductions*. *Verification: Theory and Practice 2003*], [H. Comon-Lundh, V. Shmatikov: *Intruder Deductions, Constraint Solving and Insecurity Decision in Presence of Exclusive or*. *LICS 2003*], [Y. Chevalier, R. Küsters, M. Rusinowitch, M. Turuani: *Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents*. *FSTTCS 2003*], [Y. Chevalier, R. Küsters, M. Rusinowitch, M. Turuani: *An NP Decision Procedure for Protocol Insecurity with XOR*. *LICS 2003*], [M. Abadi, V. Cortier: *Deciding Knowledge in Security Protocols Under Equational Theories*. *ICALP 2004*], [C. Lynch, C. Meadows: *Sound Approximations to Diffie-Hellman Using Rewrite Rules*. *ICICS 2004*], [M. Abadi, V. Cortier: *Deciding Knowledge in Security Protocols under (Many More) Equational Theories*. *CSFW 2005*], [H. Comon-Lundh, S. Delaune: *The Finite Variant Property: How to Get Rid of Some Algebraic Properties*. *RTA 2005*], [P. Lafourcade, D. Lugiez, R. Treinen: *Intruder Deduction for AC-Like Equational Theories with Homomorphisms*. *RTA 2005*], [J. Millen, V. Shmatikov: *Symbolic protocol analysis with an Abelian group operator or Diffie-Hellman exponentiation*. *Journal of Computer Security 13(3)*], [S. Delaune, P. Lafourcade, D. Lugiez, R. Treinen: *Symbolic Protocol Analysis in Presence of a Homomorphism Operator and Exclusive Or*. *ICALP (2) 2006*], [M. Abadi, V. Cortier: *Deciding knowledge in security protocols under equational theories*. *Theor. Comput. Sci. 367(1-2)*], [S. Escobar, C. Meadows, J. Meseguer: *Equational Cryptographic Reasoning in the Maude-NRL Protocol Analyzer*. *ENTCS 171(4)*], [S. Bursuc, H. Comon-Lundh, S. Delaune: *Associative-Commutative Deducibility Constraints*. *STACS 2007*], [P. Lafourcade, D. Lugiez, R. Treinen: *Intruder deduction for the equational theory of Abelian groups with distributive encryption*. *Inf. Comput. 205(4)*], [M. Arnaud, V. Cortier, S. Delaune: *Combining Algorithms for Deciding Knowledge in Security Protocols*. *FroCos 2007*], [V. Cortier, S. Delaune: *Deciding Knowledge in Security Protocols for Monoidal Equational Theories*. *LPAR 2007*]

Are these abstractions **sound**?

A model with equational theories

- ▶ arbitrary signature $(\mathcal{S}, \mathcal{F})$

$T ::=$		term of sort $s \in \mathcal{S}$
	x	variable x of sort $s \in \mathcal{S}$
	a	name a of sort $s \in \mathcal{S}$
	$f(T_1, \dots, T_k)$	application of symbol $f \in \mathcal{F}$

- ▶ arbitrary equational theories

Example

$$E_{\text{enc}} = \{\text{dec}(\text{enc}(x, y), y) = x, \text{enc}(\text{dec}(x, y), y) = x\}$$

$$E_{\oplus} = \{x \oplus y = y \oplus x, x \oplus (y \oplus z) = (x \oplus y) \oplus z, x \oplus 0 = x, x \oplus x = 0\}$$

A concrete implementation

A $(\mathcal{S}, \mathcal{F})$ -computational algebra A consists of

- ▶ a non-empty set of bit-strings $\llbracket s \rrbracket_A \subseteq \{0, 1\}^*$ for each $s \in \mathcal{S}$;
- ▶ a computable function $\llbracket f \rrbracket_A : \llbracket s_1 \rrbracket_A \times \dots \times \llbracket s_k \rrbracket_A \rightarrow \llbracket s \rrbracket_A$ for each $f \in \mathcal{F}$ with $\text{ar}(f) = s_1 \times \dots \times s_k \rightarrow s$;
- ▶ a computable congruence $=_{A,s}$ for each sort s in order to check the equality of elements in $\llbracket s \rrbracket_A$
- ▶ an effective procedure to draw random elements from $\llbracket s \rrbracket_A$

We associate to each formal term T a **distribution** $\llbracket T \rrbracket_A$

- ▶ $\llbracket a \rrbracket_A$ is a random value in $\llbracket s \rrbracket_A$ if a is a name of sort s
- ▶ $\llbracket f(T_1, \dots, T_n) \rrbracket_A = \llbracket f \rrbracket_A(\llbracket T_1 \rrbracket_A, \dots, \llbracket T_n \rrbracket_A)$

Families of computational algebras and distributions indexed by a security parameter η

Frames and deducibility

Terms are organized into **frames**:

$$\varphi_1 = \{x_1 \mapsto \text{enc}(k_1, k_2), x_2 \mapsto \text{enc}(k_4, k_3), x_3 \mapsto k_3\}$$

A frame reflects the view of an adversary

- ▶ the variables x_i are the (closed) terms observed by the adversary
- ▶ the names k_i are all fresh names, i.e. secret

Frames and deducibility

Terms are organized into **frames**:

$$\varphi_1 = \{x_1 \mapsto \text{enc}(k_1, k_2), x_2 \mapsto \text{enc}(k_4, k_3), x_3 \mapsto k_3\}$$

A frame reflects the view of an adversary

- ▶ the variables x_i are the (closed) terms observed by the adversary
- ▶ the names k_i are all fresh names, i.e. secret

Definition (Formal deducibility)

$\varphi \vdash_E T$ if there exists a term M with $\text{names}(M) \cap \text{names}(\varphi) = \emptyset$ such that $M\varphi =_E T$.

Example

- ▶ k_4 is deducible from φ_1 since $\text{dec}(x_2, x_3)\varphi_1 =_E k_4$;
- ▶ but neither k_1 nor k_2 are deducible.

Static Equivalence

Definition (static equivalence)

$\varphi_1 \approx_E \varphi_2$ if

- ▶ $\text{dom}(\varphi_1) = \text{dom}(\varphi_2)$
- ▶ for all M, N with $\text{names}(M, N) \cap \text{names}(\varphi) = \emptyset$, we have:
 $M\varphi_1 =_E N\varphi_1$ iff $M\varphi_2 =_E N\varphi_2$.

Intuitively, an adversary cannot find an **equation** that **distinguishes** the two frames, i.e. holds for one frame, but not for the other one.

Example

$$\phi_1 = \{x \mapsto \text{enc}(0, k)\}$$

$$\phi_3 = \{x \mapsto \text{enc}(0, k), y \mapsto k\}$$

$$\phi_2 = \{x \mapsto \text{enc}(1, k)\}$$

$$\phi_4 = \{x \mapsto \text{enc}(0, k'), y \mapsto k\}$$

Static Equivalence

Definition (static equivalence)

$\varphi_1 \approx_E \varphi_2$ if

- ▶ $\text{dom}(\varphi_1) = \text{dom}(\varphi_2)$
- ▶ for all M, N with $\text{names}(M, N) \cap \text{names}(\varphi) = \emptyset$, we have:
 $M\varphi_1 =_E N\varphi_1$ iff $M\varphi_2 =_E N\varphi_2$.

Intuitively, an adversary cannot find an **equation** that **distinguishes** the two frames, i.e. holds for one frame, but not for the other one.

Example

$$\begin{array}{ll} \phi_1 = \{x \mapsto \text{enc}(0, k)\} & \approx_E \quad \phi_2 = \{x \mapsto \text{enc}(1, k)\} \\ \phi_3 = \{x \mapsto \text{enc}(0, k), y \mapsto k\} & \not\approx_E \quad \phi_4 = \{x \mapsto \text{enc}(0, k'), y \mapsto k\} \end{array}$$

Static Equivalence: tool support

There exists **exact** [Abadi,Cortier] and **approximate** [Blanchet] algorithms for deciding static equivalence for many theories.

Tools have been implemented for verifying static equivalence

- ▶ YAPA [Baudet, Cortier, Delaune]
- ▶ KISS [Ciobâcă, Delaune, Kremer]
- ▶ Proverif [Blanchet]

Soundness and Faithfulness

Definition (Soundness and faithfulness)

A family of computational algebras (A_η) is:

- ▶ **$=_E$ -sound** iff $T_1 =_E T_2$ implies $\mathbb{P} \left[e_1, e_2 \stackrel{R}{\leftarrow} \llbracket T_1, T_2 \rrbracket_{A_\eta} : e_1 =_{A_\eta} e_2 \right]$ is overwhelming
- ▶ **$=_E$ -faithful** iff $T_1 \neq_E T_2$ implies $\mathbb{P} \left[e_1, e_2 \stackrel{R}{\leftarrow} \llbracket T_1, T_2 \rrbracket_{A_\eta} : e_1 =_{A_\eta} e_2 \right]$ is negligible
- ▶ **\approx_E -sound** iff $\varphi_1 \approx_E \varphi_2$ implies $\llbracket \varphi_1 \rrbracket_{A_\eta} \approx \llbracket \varphi_2 \rrbracket_{A_\eta}$
- ▶ **\approx_E -faithful** iff $\varphi_1 \not\approx_E \varphi_2$ implies there exists PPT \mathcal{A} , $\text{Adv}^{\text{IND}}(\mathcal{A}, \eta, \llbracket \varphi_1 \rrbracket_{A_\eta}, \llbracket \varphi_2 \rrbracket_{A_\eta})$ is overwhelming
- ▶ **$\not\vdash_E$ -sound** iff $\varphi \not\vdash_E T$ implies that for every PPT \mathcal{A} , $\mathbb{P} \left[\phi, e \stackrel{R}{\leftarrow} \llbracket \varphi, T \rrbracket_{A_\eta} : \mathcal{A}(\phi) =_{A_\eta} e \right]$ is negligible
- ▶ **$\not\vdash_E$ -faithful** iff $\varphi \vdash_E T$ implies that there exists a PPT \mathcal{A} , $\mathbb{P} \left[\phi, e \stackrel{R}{\leftarrow} \llbracket \varphi, T \rrbracket_{A_\eta} : \mathcal{A}(\phi) =_{A_\eta} e \right]$ is overwhelming

Unconditional Soundness

Definition (Unconditional soundness)

A family of computational algebras (A_η) is:

- ▶ **unconditionally $=_E$ -sound** iff $T_1 =_E T_2$ implies
$$\mathbb{P} \left[e_1, e_2 \stackrel{R}{\leftarrow} \llbracket T_1, T_2 \rrbracket_{A_\eta} : e_1 =_{A_\eta} e_2 \right] = 1$$
- ▶ **unconditionally \approx_E -sound** iff $\varphi_1 \approx_E \varphi_2$ implies $\llbracket \varphi_1 \rrbracket = \llbracket \varphi_2 \rrbracket$
- ▶ **unconditionally $\not\approx_E$ -sound** iff $\varphi \not\approx_E T$ implies that the drawings for φ and T are independent, i.e., for all ϕ_0, e_0 ,
$$\mathbb{P} \left[\phi_0, e_0 \stackrel{R}{\leftarrow} \llbracket \varphi, T \rrbracket_{A_\eta} \right] = \mathbb{P} \left[\phi_0 \stackrel{R}{\leftarrow} \llbracket \varphi \rrbracket_{A_\eta} \right] \times \mathbb{P} \left[e_0 \stackrel{R}{\leftarrow} \llbracket T \rrbracket_{A_\eta} \right]$$

Some Properties

Proposition 1

- ▶ $=_E$ -soundness implies $\not\approx_E$ -faithfulness
- ▶ $=_E$ -soundness and $=_E$ -faithfulness implies \approx_E -faithfulness

Proposition 2

Assume that free binary symbols $h_s : s \times \text{Key} \rightarrow \text{Hash}$ are available for every sort s .

- ▶ Then \approx_E -soundness implies $=_E$ -faithfulness and $\not\approx_E$ -soundness
- ▶ if the implementations for the h_s are collision-resistant, then \approx_E -soundness also implies $=_E$ -soundness, \approx_E -faithfulness and, $\not\approx_E$ -faithfulness

A \approx_E -soundness Criterion

Standard implementations: unc. $=_E$ -sound + uniform distributions

Definition (Ideal semantics)

$\llbracket \varphi \rrbracket^{ideal}$ is the uniform distribution over

$$\{ \{x_1 \mapsto e_1, \dots, x_n \mapsto e_n\} \mid (e_1, \dots, e_n) \in \llbracket s_1 \rrbracket \times \dots \times \llbracket s_n \rrbracket \text{ and} \\ \forall (M, N), (M\varphi =_E N\varphi) \Rightarrow \llbracket M \rrbracket_{\{x_1 \mapsto e_1, \dots, x_n \mapsto e_n\}} = \llbracket N \rrbracket_{\{x_1 \mapsto e_1, \dots, x_n \mapsto e_n\}} \}$$

Theorem

Assume that for any φ , $\llbracket \varphi \rrbracket^{ideal} \approx \llbracket \varphi \rrbracket$.

Then we have \approx_E -soundness.

Patterns and transparent frames

Abadi and Rogaway: encryptions for which the key is unknown are replaced by \square . **Transparent frames** are a general concept inspired by patterns.

Definition (Transparent frames)

A frame φ is **transparent** if all its subterms are deducible. A theory E is transparent if for any frame φ , there exists a transparent frame $\bar{\varphi}$ s.t. $\varphi \approx_E \bar{\varphi}$

Example

$$\varphi = \{x_1 \mapsto n_1 \oplus n_2, x_2 \mapsto n_2 \oplus n_3, x_3 \mapsto n_1 \oplus n_3\}.$$

There are several transparent frames equivalent to φ , for instance

$$\begin{aligned} &\{x_1 \mapsto n_1 \oplus n_2, x_2 \mapsto n_1, x_3 \mapsto n_2\}, \\ &\{x_1 \mapsto n_1, x_2 \mapsto n_1 \oplus n_2, x_3 \mapsto n_2\} \text{ and} \\ &\{x_1 \mapsto n_1, x_2 \mapsto n_2, x_3 \mapsto n_1 \oplus n_2\}. \end{aligned}$$

Necessity of the soundness criterion

Proposition

Assume a standard implementation. Let φ be a transparent frame. Then $\llbracket \varphi \rrbracket = \llbracket \varphi \rrbracket^{ideal}$.

Corollary

Our criterion is necessary for transparent theories. Assume a standard implementation and that E is transparent. Then \approx_E -soundness implies that for any φ , $\llbracket \varphi \rrbracket^{ideal} \approx \llbracket \varphi \rrbracket$.

Part V

Sound equational theories

Application 1: Exclusive Or

$$E_{\oplus} : \quad \begin{array}{l} x \oplus y = y \oplus x \\ (x \oplus y) \oplus z = x \oplus (y \oplus z) \end{array} \quad \begin{array}{l} x \oplus x = 0 \\ x \oplus 0 = x \end{array}$$

Theorem

The usual implementation for the XOR theory is

- ▶ *unconditionally $=_{E_{\oplus}}$ -, $\approx_{E_{\oplus}}$ - and $\not\vdash_{E_{\oplus}}$ -sound.*
- ▶ *It is also $=_{E_{\oplus}}$ -, $\approx_{E_{\oplus}}$ - and $\not\vdash_{E_{\oplus}}$ -faithful.*

Application 2: Ciphers

Ciphers are deterministic, length-preserving symmetric encryption functions.

E_{sym} :

$$\begin{array}{ll} \text{dec}_n(\text{enc}_n(x, y), y) = x & \text{cons}_n(\text{head}_n(x), \text{tail}_n(x)) = x \\ \text{enc}_n(\text{dec}_n(x, y), y) = x & \text{enc}_0(\text{nil}, x) = \text{nil} \\ \text{head}_n(\text{cons}_n(x, y)) = x & \text{dec}_0(\text{nil}, x) = \text{nil} \\ \text{tail}_n(\text{cons}_n(x, y)) = y & \end{array}$$

Proposition

E_{sym} is transparent.

$\bar{\varphi}$ is obtained from φ by replacing non-deducible subterms with fresh names.

Soundness of Ciphers

A frame is **well-formed** if:

- ▶ there are no **head** or **tail** symbols,
- ▶ there are only **atomic keys**
- ▶ there are no **encryption cycles**

Theorem

Assume that the concrete implementations for the encryption and its inverse satisfy both the ω -IND-P1-C1 assumption. Let φ_1 and φ_2 be two well-formed frames.

If $\varphi_1 \approx_{E_{\text{sym}}} \varphi_2$ then $\llbracket \varphi_1 \rrbracket \approx \llbracket \varphi_2 \rrbracket$.

Note that two transparent, statically equivalent frames are not necessarily equal up to renaming, e.g.

$$\{x \mapsto \text{enc}(k_1, k_2), y \mapsto k_2\} \approx_{E_{\text{sym}}} \{x \mapsto \text{dec}(k_1, k_2), y \mapsto k_2\}$$

Proof is simplified by the use of ideal semantics.

Application 3: Modular exponentiation

Consider the following signature modelling modular exponentiation

$\text{exp} : R \rightarrow G$	exponentiation	$+, \cdot : R \times R \rightarrow R$	add, mult
$*$: $G \times G \rightarrow G$	mult in G	$- : R \rightarrow R$	inverse
		$0_R, 1_R : R$	constants

and the equational theory E_{DH}

$$x + y = y + x$$

$$(x + y) + z = x + (y + z)$$

$$(x \cdot y) \cdot z = x \cdot (y \cdot z)$$

$$0_R + x = x$$

$$x \cdot y = y \cdot x$$

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

$$x + (-x) = 0_R$$

$$1_R \cdot x = x$$

$$\text{exp}(x) * \text{exp}(y) = \text{exp}(x + y)$$

Some restrictions on frames:

- ▶ only elements of sort G
- ▶ products have to be power-free, i.e. x^n is forbidden for $n > 1$
- ▶ products must not contain more than ℓ elements for a fixed bound ℓ

Soundness of modular exponentiation

Concrete implementation

- ▶ Instance Generator IG : a polynomial-time algorithm that outputs a cyclic group \mathcal{G} of prime order q
- ▶ Operations are the usual implementations over \mathbb{Z}_q

An instance generator satisfies the **DDH assumption** if

$\left| \mathbb{P} \left[(g, q) \leftarrow IG(\eta) : a, b \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^{ab}) = 1 \right] - \mathbb{P} \left[(g, q) \leftarrow IG(\eta) : a, b, c \leftarrow \mathbb{Z}_q : \mathcal{A}(g^a, g^b, g^c) = 1 \right] \right|$ is negligible.

Theorem

(A_η) is $\approx_{E_{\text{DH}}}$ -sound iff (A_η) satisfies the **DDH assumption**.

The proof uses the 3DH assumption of [BressonLakhnechMazaréWarinski].

Application 4: Offline guessing attacks [AbadiBaudetWarinschi]

Offline guessing attacks have an elegant formulation in terms of static equivalence

Definition (Resistance against offline guessing attacks)

A frame φ is resistant against offline guessing attacks on w iff

$$\varphi \mid \{x \mapsto w\} \approx \varphi \mid \{x \mapsto w'\}$$

where $x \notin \text{dom}(\varphi)$ and $w' \notin \text{names}(\varphi)$.

Application 4: Offline guessing attacks [AbadiBaudetWarinski]

Offline guessing attacks have an elegant formulation in terms of static equivalence

Definition (Resistance against offline guessing attacks)

A frame φ is resistant against offline guessing attacks on w iff

$$\varphi \mid \{x \mapsto w\} \approx \varphi \mid \{x \mapsto w'\}$$

where $x \notin \text{dom}(\varphi)$ and $w' \notin \text{names}(\varphi)$.

Soundness result for a rich theory including

- ▶ ciphers for encryption with passwords
- ▶ symmetric encryption
- ▶ public-key encryption

Soundness of static equivalence implies resistance against a computational definition of guessing attacks

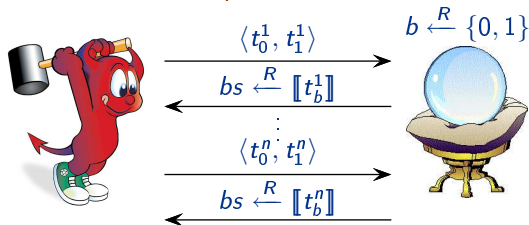
Part VI

Adaptive adversaries

Adaptive Soundness of Static Equivalence

[MicciancioPanjwani05]: soundness in the presence of an adaptive adversary for an Abadi-Rogaway-like model

Adaptive soundness of static equivalence



A_η is \approx_E -ad-sound if for any sequence $\langle t_0^1, t_1^1 \rangle, \langle t_0^2, t_1^2 \rangle, \dots, \langle t_0^n, t_1^n \rangle$ we have

$$\{x_1 \mapsto t_0^1, x_2 \mapsto t_0^2, \dots, x_n \mapsto t_0^n\} \approx_E \{x_1 \mapsto t_0^1, x_2 \mapsto t_0^2, \dots, x_n \mapsto t_0^n\}$$

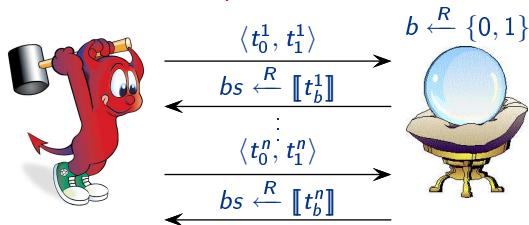
implies

$$\text{Adv}_{\mathcal{A}, A_\eta}^{\text{ADPT}}(\eta) = \mathbb{P} \left[\mathcal{A}^{\mathcal{O}_{LR, A_\eta}^1} = 1 \right] - \mathbb{P} \left[\mathcal{A}^{\mathcal{O}_{LR, A_\eta}^0} = 1 \right] \text{ is negligible in } \eta$$

Adaptive Soundness of Static Equivalence

[MicciancioPanjwani05]: soundness in the presence of an adaptive adversary for an Abadi-Rogaway-like model

Adaptive soundness of static equivalence



A_η is **unconditionally** \approx_E -**ad-sound** if for any sequence

$\langle t_0^1, t_1^1 \rangle, \langle t_0^2, t_1^2 \rangle, \dots, \langle t_0^n, t_1^n \rangle$ we have

$$\{x_1 \mapsto t_0^1, x_2 \mapsto t_0^2, \dots, x_n \mapsto t_0^n\} \approx_E \{x_1 \mapsto t_0^1, x_2 \mapsto t_0^2, \dots, x_n \mapsto t_0^n\}$$

implies

$$\text{Adv}_{\mathcal{A}, A_\eta}^{\text{ADPT}}(\eta) = \mathbb{P} \left[\mathcal{A}^{\mathcal{O}_{LR, A_\eta}^1} = 1 \right] - \mathbb{P} \left[\mathcal{A}^{\mathcal{O}_{LR, A_\eta}^0} = 1 \right] \text{ is } 0$$

Adaptive soundness is strictly stronger!

Proposition

If A_η is \approx_E -ad-sound then A_η is also \approx_E -sound but the converse is false in general.

Proof idea: Consider the following signature without any equations

0, 1 : Bit
cons : Bit \times BS \rightarrow BS
eq : BS \times Nonce \rightarrow Bool

$\llbracket eq \rrbracket(bs, N)$ outputs 1 if $bs = N$, 0 otherwise. We do have \approx -soundness, but not \approx -ad-soundness

Proposition

A_η is unconditionally \approx_E -ad-sound iff A_η is unconditionally \approx_E -sound.

An example: Symmetric encryption

Consider the following signature modelling symmetric encryption

$$\begin{array}{ll} \text{enc, dec} & : \text{Data} \times \text{Data} \rightarrow \text{Data} & \text{samekey} & : \text{Data} \times \text{Data} \rightarrow \text{Data} \\ \text{pair} & : \text{Data} \times \text{Data} \rightarrow \text{Data} & \text{tenc, tpair} & : \text{Data} \rightarrow \text{Data} \\ \pi_l, \pi_r & : \text{Data} \rightarrow \text{Data} & 0, 1 & : \text{Data} \end{array}$$

and the equational theory

$$\begin{array}{ll} \text{dec}(\text{enc}(x, y), y) & = x & \text{samekey}(\text{enc}(x, y), \text{enc}(z, y)) & = 1 \\ \pi_l(\text{pair}(x, y)) & = x & \text{tenc}(\text{enc}(x, y)) & = 1 \\ \pi_r(\text{pair}(x, y)) & = y & \text{tpair}(\text{pair}(x, y)) & = 1 \end{array}$$

Restrictions on frames

- ▶ no **key cycles**
- ▶ no **destructor symbols**
- ▶ if k is used as plaintext in t_i , then k cannot be used at a key position in t_j for $j < i$ (avoids **selective decommitment**)

Soundness of symmetric encryption

Concrete implementation

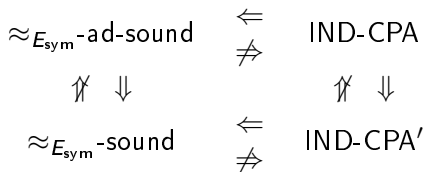
We suppose semantic secure encryption (**IND-CPA**):

$$\text{Adv}_{\mathcal{S}\mathcal{E}, \mathcal{A}}^{\text{cpa}}(\eta) = \left| \mathbb{P} \left[\mathcal{A}^{LR_{\mathcal{S}\mathcal{E}}^1}(\eta) = 1 \right] - \mathbb{P} \left[\mathcal{A}^{LR_{\mathcal{S}\mathcal{E}}^0}(\eta) = 1 \right] \right|$$

where $LR_{\mathcal{S}\mathcal{E}}^b$ is a left-right encryption oracle.

We also consider a variant **IND-CPA'**: the oracle only accepts a **single call**. Given a list of pairs of bitstrings $(\langle bs_0^i, bs_1^i \rangle)_i$ the oracle returns the list $(\mathcal{E}(bs_b^i, k))_i$.

Soundness results



Exclusive OR and Modular exponentiation

Soundness results for XOR

The usual implementation for the XOR theory is unconditional $\approx_{E_{DH}}$ -ad-sound

Soundness results for modular exponentiation

$$\begin{array}{lcl} \approx_{E_{DH}}\text{-ad-sound} & \iff & \text{DDH} \\ & & \updownarrow \\ \approx_{E_{DH}}\text{-sound} & \iff & \text{DDH} \end{array}$$

Part VII

Combination results and constructed keys

Combining signatures

Definition (Disjoint signatures)

Let $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1)$ and $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2)$ are **disjoint** iff $\mathcal{F}_1 \cap \mathcal{F}_2 = \emptyset$ and $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$.

Definition (Signature combination, Layered signatures)

Let $\Sigma_1 = (\mathcal{S}_1, \mathcal{F}_1)$ and $\Sigma_2 = (\mathcal{S}_2, \mathcal{F}_2)$ be two disjoint signatures. We say that a subsort relation S is a **signature combination** for Σ_1 and Σ_2 if $S \subseteq \mathcal{S}_2 \times \mathcal{S}_1$.

$\Sigma = (\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{F}_1 \cup \mathcal{F}_2)$ is a $(\Sigma_1, \Sigma_2)_S$ -layered signature.

Example (Encryption and pseudo-random generators)

Let $\Sigma_1 = (\{\text{Data}\}, \{\text{enc}, \text{dec}\})$ and $\Sigma_2 = (\{\text{Rand}\}, \{\text{prg}\})$ and $\text{Rand } S \text{ Data}$. Then $\Sigma_1 \cup \Sigma_2$ is $(\Sigma_1, \Sigma_2)_S$ -layered.

Hybrid functions

Let $\Sigma = \Sigma_1 \cup \Sigma_2$ be a $(\Sigma_1, \Sigma_2)_S$ -layered signature equipped with equational theories E_1 and E_2 .

A (E_1, E_2) -**hybrid function** for a set F of pairs of frames is a function σ from lists of terms over Σ to terms over Σ such that:

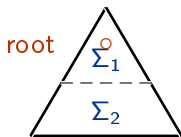
- ▶ for any frame φ occurring in F , $\varphi \approx_{E_1} \sigma(\varphi)$ where $\sigma(\{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}) = \{x_1 \mapsto \sigma([t_1]), \dots, x_n \mapsto \sigma([t_1 \dots t_n])\}$;
- ▶ for any $(\varphi, \varphi') \in F$, if $\varphi \approx_{E_1 \cup E_2} \varphi'$ then let $\varphi = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ and $\varphi' = \{x_1 \mapsto u_1, \dots, x_n \mapsto u_n\}$. $\forall 1 \leq i \leq n$

Hybrid functions

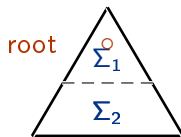
Let $\Sigma = \Sigma_1 \cup \Sigma_2$ be a $(\Sigma_1, \Sigma_2)_S$ -layered signature equipped with equational theories E_1 and E_2 .

A (E_1, E_2) -**hybrid function** for a set F of pairs of frames is a function σ from lists of terms over Σ to terms over Σ such that:

- ▶ for any frame φ occurring in F , $\varphi \approx_{E_1} \sigma(\varphi)$ where $\sigma(\{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}) = \{x_1 \mapsto \sigma([t_1]), \dots, x_n \mapsto \sigma([t_1 \dots t_n])\}$;
- ▶ for any $(\varphi, \varphi') \in F$, if $\varphi \approx_{E_1 \cup E_2} \varphi'$ then let $\varphi = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ and $\varphi' = \{x_1 \mapsto u_1, \dots, x_n \mapsto u_n\}$. $\forall 1 \leq i \leq n$
 - ▶ the **roots** of all Σ_1 positions in $\sigma([t_1 \dots t_i])$ and $\sigma([u_1 \dots u_i])$ coincide



=

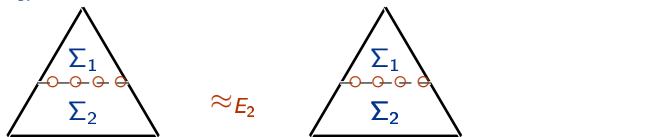


Hybrid functions

Let $\Sigma = \Sigma_1 \cup \Sigma_2$ be a $(\Sigma_1, \Sigma_2)_S$ -layered signature equipped with equational theories E_1 and E_2 .

A (E_1, E_2) -**hybrid function** for a set F of pairs of frames is a function σ from lists of terms over Σ to terms over Σ such that:

- ▶ for any frame φ occurring in F , $\varphi \approx_{E_1} \sigma(\varphi)$ where $\sigma(\{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}) = \{x_1 \mapsto \sigma([t_1]), \dots, x_n \mapsto \sigma([t_1 \dots t_n])\}$;
- ▶ for any $(\varphi, \varphi') \in F$, if $\varphi \approx_{E_1 \cup E_2} \varphi'$ then let $\varphi = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$ and $\varphi' = \{x_1 \mapsto u_1, \dots, x_n \mapsto u_n\}$. $\forall 1 \leq i \leq n$
 - ▶ the **roots** of all Σ_1 positions in $\sigma([t_1 \dots t_i])$ and $\sigma([u_1 \dots u_i])$ **coincide**
 - ▶ the frames of all minimal Σ_2 positions in $\sigma([t_1 \dots t_i])$ and $\sigma([u_1 \dots u_i])$ are E_2 **statically equivalent**



Combining soundness results

Proposition 3

We consider the families of computational algebras (A_η^1) for Σ_1 and (A_η^2) for Σ_2 respecting S , i.e. $(s_2, s_1) \in S$ implies that $\llbracket s_2 \rrbracket_{A_\eta^2} \subseteq \llbracket s_1 \rrbracket_{A_\eta^1}$.

Let F be a set of pair of frames over $\Sigma_1 \cup \Sigma_2$ and σ be a (E_1, E_2) -hybrid function. If $A_\eta^1 \times A_\eta^2$ is \approx_{E_1} -ad-sound for $G = \{(\varphi, \sigma(\varphi)) \mid \varphi \text{ occurs in } F\}$ and A_η^2 is \approx_{E_2} -ad-sound for frames on Σ_2 , then $A_\eta^1 \times A_\eta^2$ is $\approx_{E_1 \cup E_2}$ -ad-sound for F .

Proof idea: Let \mathcal{A} be an adversary against $E_1 \cup E_2$ -ad-soundness that queries his oracle with a pair of frames (φ, φ') . Build

- ▶ \mathcal{B}_1 : adversary against E_1 -ad-soundness that queries $(\varphi, \sigma(\varphi))$
- ▶ \mathcal{B}_2 : adversary against E_2 -ad-soundness that queries $(\sigma(\varphi), \sigma(\varphi'))$
- ▶ \mathcal{B}_3 : adversary against E_1 -ad-soundness that queries $(\sigma(\varphi), \varphi')$

The advantage of \mathcal{A} is the sum of the advantages of \mathcal{B}_1 , \mathcal{B}_2 and \mathcal{B}_3 .

Combination of symmetric encryption and modular exponentiation

We show adaptive soundness for a combination of previous theories

- ▶ a soundness result for **symmetric encryption + modular exponentiation**

we suppose that sort G is a subsort of **Data**:

- ▶ group elements can appear **inside encryptions** or at **key positions**

$\{x_1 \mapsto \text{exp}(a), x_2 \mapsto \text{enc}(\text{exp}(a) * \text{exp}(b), \text{exp}(a))\}$ is valid

$\{x_1 \mapsto \text{exp}(a), x_2 \mapsto \text{enc}(\text{exp}(a), \text{exp}(b)) * \text{enc}(\text{exp}(a), \text{exp}(c))\}$
is invalid

- ▶ the IND-CPA scheme uses **group elements as keys**, together with a **randomness extractor**

Combination of symmetric encryption and XOR

Similarly we show adaptive soundness for for **symmetric encryption + XOR**

We suppose that sort Data_\oplus is a subsort of Data :

- ▶ terms of sort Data_\oplus can appear **inside encryptions** or at **key positions**
- ▶ the key generation of the IND-CPA scheme randomly samples elements of $\{0, 1\}^\eta$

Part VIII

DKE: an application of adaptive adversaries

An application: dynamic group key exchange protocols

A protocol is specified by four functions $(\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})$:

- ▶ $\mathcal{S}(U_1, \dots, U_n)$: **setup** for users (U_1, \dots, U_n) returns terms modelling the execution of the setup protocol
- ▶ $\mathcal{J}(U), \mathcal{L}(U)$: **join**, **leave** of user U returns the terms modelling the execution of the join/leave sub-protocol
- ▶ \mathcal{K} : the **key** function returns the current group key

We consider **static corruption** (once at the beginning of the protocol).

In the symbolic model, the above functions yield a transition system associating the frame of exchanged messages to each state.

A DKE protocol is **symbolically secure** if for any reachable state s

$$\phi(s) \cup \{x \mapsto \mathcal{K}(s)\} \approx_E \phi(s) \cup \{x \mapsto r\}$$

Soundness result for DKE

In the concrete model, we suppose an **oracle** modeling **corruption**, \mathcal{S} , \mathcal{J} and \mathcal{L} .

Moreover, there is a **final Test call** which returns either the key or a random key (depending on the challenge bit).

A DKE protocol is **secure in the concrete model** if the advantage

$$\text{Adv}_{\mathcal{A}, \mathcal{A}_\eta}^{(\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})}(\eta) = \mathbb{P}[\mathcal{A}^{\mathcal{O}_1} = 1] - \mathbb{P}[\mathcal{A}^{\mathcal{O}_0} = 1]$$

is negligible for any adversary.

Soundness result for DKE and modular exponentiation

Let (\mathcal{A}_η) be a family of computational algebras and $\mathcal{SE} = (\mathcal{S}, \mathcal{J}, \mathcal{L}, \mathcal{K})$ be a DKE. If (\mathcal{A}_η) is $\approx_{E_{\text{DH}}}$ -ad-sound and \mathcal{SE} is secure in the symbolic model, then \mathcal{SE} is secure in the concrete model.

Some references on soundness of static equivalence

[BCK05] M. Baudet, V. Cortier, and S. Kremer. Computationally sound implementations of equational theories against passive adversaries. ICALP'05, long version in Information and Computation.

[ABW06] M. Abadi, M. Baudet, and B. Warinschi. Guessing attacks and the computational soundness of static equivalence. In FoSSaCS'06.

[KM07] S. Kremer and L. Mazaré. Adaptive soundness of static equivalence. In ESORICS'07.

Part IX

Active adversaries: trace mapping

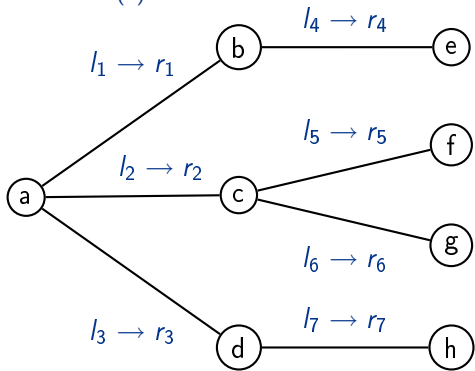
Formal Model

- ▶ agents A_i, a_i , nonces $X_{A_i}^j, n(a_i, j, s)$, garbage G
- ▶ pairing $\langle m_1, m_2 \rangle$
- ▶ asymmetric encryption $\{m\}_{\text{ek}(a)}^!$

Formal Model

- ▶ agents A_i, a_i , nonces $X_{A_i}^j, n(a_i, j, s)$, garbage G
- ▶ pairing $\langle m_1, m_2 \rangle$
- ▶ asymmetric encryption $\{m\}_{ek(a)}$

A **protocol** is a finite set of **roles** of the form:



Example

Needham-Schroeder-Lowe protocol

$$A \rightarrow B : \quad \{N_a, A\}_{\text{ek}(B)}$$

$$B \rightarrow A : \quad \{N_a, N_b, B\}_{\text{ek}(A)}$$

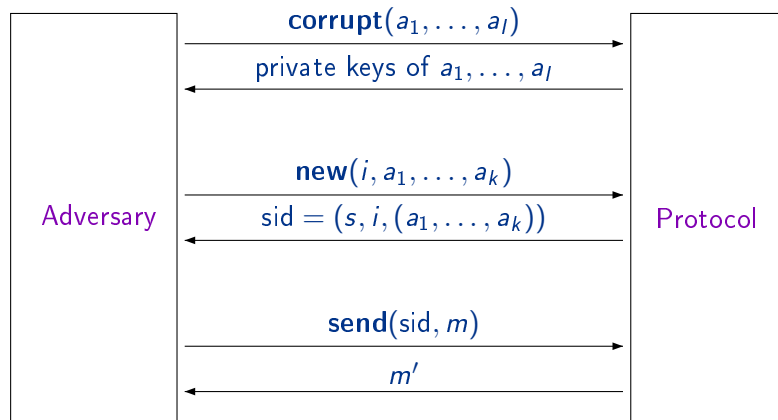
$$A \rightarrow B : \quad \{N_b\}_{\text{ek}(B)}$$

$$\begin{aligned} \Pi(1) = & (\text{init} \rightarrow \{X_{A_1}^1, A_1\}_{\text{ek}(A_2)}^{\text{ag}(1)}), \\ & (\{X_{A_1}^1, X_{A_2}^1, A_2\}_{\text{ek}(A_1)}^L \rightarrow \{X_{A_2}^1\}_{\text{ek}(A_2)}^{\text{ag}(1)}) \end{aligned}$$

$$\begin{aligned} \Pi(2) = & (\{X_{A_1}^1, A_1\}_{\text{ek}(A_2)}^{L_1} \rightarrow \{X_{A_1}^1, X_{A_2}^1, A_2\}_{\text{ek}(A_1)}^{\text{ag}(1)}), \\ & (\{X_{A_2}^1\}_{\text{ek}(A_2)}^{L_2} \rightarrow \text{stop}) \end{aligned}$$

Variables are local to each role and each session.

Network



Formal Intruder Deduction Rules and Implementation

$$\frac{}{\phi \vdash m} \quad m \in \phi \quad \frac{\phi \vdash b}{\phi \vdash \text{ek}(b)} \quad b \in A \cup X.a$$

$$\frac{\phi \vdash m_1 \quad \phi \vdash m_2}{\phi \vdash \langle m_1, m_2 \rangle} \quad \frac{\phi \vdash \langle m_1, m_2 \rangle}{\phi \vdash m_i} \quad i \in \{1, 2\}$$

$$\frac{\phi \vdash \text{ek}(b), \phi \vdash m}{\phi \vdash \{m\}_{\text{ek}(b)}^r} \quad r \in \text{rand} \quad \frac{\phi \vdash \{m\}_{\text{ek}(b)}^r \quad \phi \vdash \text{dk}(b)}{\phi \vdash m}$$

► encryption : IND-CCA2

→ the adversary cannot distinguish between $\{n_0\}_k$ and $\{n_1\}_k$ even if he has access to encryption and decryption oracles.

Formal Intruder Deduction Rules and Implementation

$$\frac{}{\phi \vdash m} \quad m \in \phi \quad \frac{\phi \vdash b}{\phi \vdash \text{ek}(b)} \quad b \in A \cup X.a$$

$$\frac{\phi \vdash m_1 \quad \phi \vdash m_2}{\phi \vdash \langle m_1, m_2 \rangle} \quad \frac{\phi \vdash \langle m_1, m_2 \rangle}{\phi \vdash m_i} \quad i \in \{1, 2\}$$

$$\frac{\phi \vdash \text{ek}(b), \phi \vdash m}{\phi \vdash \{m\}_{\text{ek}(b)}^r} \quad r \in \text{rand} \quad \frac{\phi \vdash \{m\}_{\text{ek}(b)}^r \quad \phi \vdash \text{dk}(b)}{\phi \vdash m}$$

► encryption : IND-CCA2

→ the adversary cannot distinguish between $\{n_0\}_k$ and $\{n_1\}_k$ even if he has access to encryption and decryption oracles.

► parsing :

- each bit-string has a label which indicates his type (identity, nonce, key, ...)
- one can retrieve the (public) encryption key from an encrypted message.

Trace mapping result

$t^s \preceq t^c$ if there exists a partial, injective function $c : M \rightarrow \mathcal{C}^\eta$ such that $t^c = c(t^s)$

$\text{Exec}^s(\Pi)$: the set of valid symbolic executions of protocol π (all terms sent by the attacker are deducible)

$\text{Exec}_{\Pi(R_\Pi), \mathcal{A}(R_A)}^c(\eta)$: concrete execution induced by adversary \mathcal{A} with adv. coins R_A , interacting with protocol Π and prot. coins R_Π

Lemma

If the encryption scheme \mathcal{AE} is IND-CCA-2 secure then for any p.p.t. \mathcal{A}

$$\Pr \left[\exists t^s \in \text{Exec}^s(\Pi) \mid t^s \preceq \text{Exec}_{\Pi(R_\Pi), \mathcal{A}(R_A)}^c(\eta) \right] \geq 1 - \nu_{\mathcal{A}}(\eta)$$

where the probability is over $(R_\Pi, R_A) \leftarrow \{0, 1\}^{p_{\mathcal{A}}(\eta)} \times \{0, 1\}^{g_{\mathcal{A}}(\eta)}$ and $\nu_{\mathcal{A}}(\cdot)$ is some negligible function.

Proof ideas:

The proof is done in two main steps:

STEP I:

- ▶ associate to each computational trace a symbolic trace using parsing

STEP II: show this symbolic trace is valid with overwhelming probability

- ▶ characterize non-valid traces: identify all ways in which the messages output by the adversary are invalid
- ▶ construct adversary \mathcal{B} against IND-CCA-2
- ▶ \mathcal{B} simulates the protocol using the encryption oracle
- ▶ if \mathcal{A} outputs a non-valid trace then \mathcal{B} breaks IND-CCA-2

Advertisement

Joe-Kai's talk on Thursday



What is the right way to state a mapping lemma.

Consequences of the mapping lemma

The mapping lemma allows to transfer **trace properties**, e.g. authentication

Specific ad-hoc results show how to obtain **nonce secrecy** (in terms of indistinguishability)

- ▶ for asymmetric encryption IND-CCA-2 implies nonce secrecy
- ▶ for asymmetric encryption (IND-CCA-2) + hash functions (random oracle): specific symbolic criterion which is sufficient and necessary for nonce secrecy

Consequences of the mapping lemma

The mapping lemma allows to transfer **trace properties**, e.g. authentication

Specific ad-hoc results show how to obtain **nonce secrecy** (in terms of indistinguishability)

- ▶ for asymmetric encryption IND-CCA-2 implies nonce secrecy
- ▶ for asymmetric encryption (IND-CCA-2) + hash functions (random oracle): specific symbolic criterion which is sufficient and necessary for nonce secrecy

The mapping lemma can be seen as the generalization to active adversaries of **\vdash -soundness**

A natural question: can we generalize **\approx -soundness** to active adversaries?

Some references on trace mapping

Public key encryption [MW04b]

D. Micciancio and B. Warinschi. Soundness of formal encryption in the presence of active adversaries. In TCC'04.

with signatures [CW05,JLM05]

V. Cortier and B. Warinschi. Computationally sound, automated proofs for security protocols. In ESOP'05.

R. Janvier, Y. Lakhnech, and L. Mazaré. Completing the picture: Soundness of formal encryption in the presence of active adversaries. In ESOP'05.

with hash functions [CKKW06,JLM06]

V. Cortier, S. Kremer, R. Küsters, and B. Warinschi. Computationally sound symbolic secrecy in the presence of hash functions. In FSTTCS'06.

R. Janvier, Y. Lakhnech, and L. Mazaré. Computational soundness of symbolic analysis for protocols using hash functions. In IC'06

Non-malleable commitment [GGvR08]

D. Galindo, F. D. Garcia, and P. van Rossum. Computational soundness of non-malleable commitments. In ISPEC'08

Zero-knowledge proofs [BU08]

M. Backes and D. Unruh. Computational soundness of symbolic zero-knowledge proofs against active attackers. In CSF'08

Part X

Soundness of observational equivalence

Observational equivalence

In symbolic models indistinguishability for active adversaries is classically modelled as **observational equivalence** (\approx_o) in cryptographic pi calculi.

Intuitively, if $P \approx_o Q$ then P and Q behave similarly in the presence of an **arbitrary environment**.

In [Comon-Lundh - Cortier, CCS08]:

$$P \approx_o Q \longrightarrow \llbracket P \rrbracket \approx \llbracket Q \rrbracket$$

for symmetric key encryption (IND-CPA + INT-CTXT encryption, key hierarchy, honest key generation, parsing) in the applied pi calculus (deterministic processes).

Observational equivalence

Warning: the following naive statement is wrong.

Trace mapping + soundness of static equivalence implies observational equivalence

In [Comon-Lundh - Cortier, CCS08]:

Trace mapping + **tree soundness** implies observational equivalence

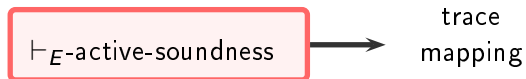
Tree soundness is an adaptive criterion on process trees, close in spirit to adaptive soundness of static equivalence, but not implied by the later.

Future work

We are still lacking a theory for computational soundness in the presence of active adversaries.

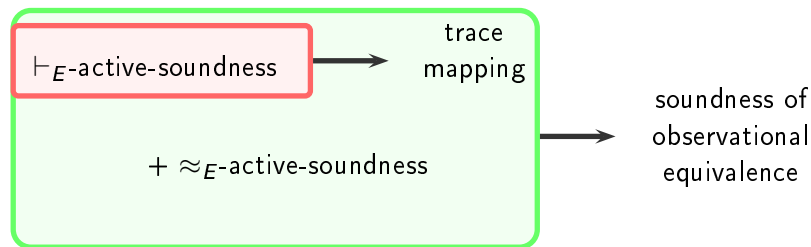
Future work

We are still lacking a theory for computational soundness in the presence of active adversaries.



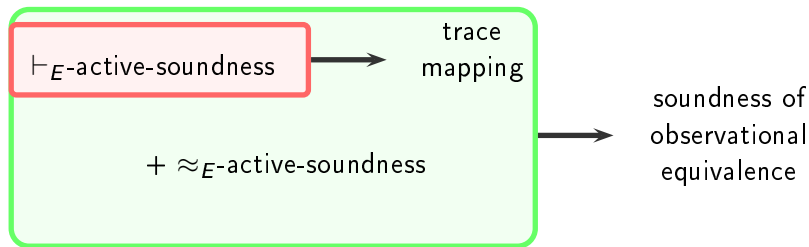
Future work

We are still lacking a theory for computational soundness in the presence of active adversaries.



Future work

We are still lacking a theory for computational soundness in the presence of active adversaries.



Independent of the particular underlying protocol specification language

Combination techniques of soundness criterion