

Guessing Attacks and the Computational Soundness of Static Equivalence

Mathieu Baudet

Joint work with Martín Abadi and Bogdan Warinschi

Formacrypt – Mar. 6, 2006

Studying cryptographic protocols

Two approaches:

- **Symbolic models**
 - + now highly automatized
 - restricted to Needham-Schroeder attackers (a.k.a. Dolev-Yao ones)
 - we may miss some attacks
- **Computational models**
 - + arbitrary PPTIME adversary
 - (so far) hand-made, complex reduction proofs

Relating the two views

- Can we justify symbolic models w.r.t. cryptography ?
→ abstract model as secure as the concrete one ?
- Main goal: **automatic proofs** of computational security

Relating the two views

- Can we justify symbolic models w.r.t. cryptography ?
→ abstract model as secure as the concrete one ?
- Main goal: **automatic proofs** of computational security

but also

- Help **clarify** “perfect” cryptographic assumptions
- Help **design** new symbolic criterion
→ e.g. guessing attacks ?!

Content

This work (to app. in [FOSSACS'06])

Security of pieces of data featuring **low-entropy** ("weak") **secrets** (e.g. passwords used as encryption keys)

- Wish to justify **static equivalence** w.r.t. cryptographic **indistinguishability** for suitable primitives and equations
- Application to guessing attacks

Extends [Abadi, Warinschi–ICALP'05] (adding **static equivalence and multiple passwords**) and [Baudet, Cortier, Kremer–ICALP'05] (adding **passwords and probabilistic encryption(s)**).

Example of guessing attack

Handshake Protocol

0. $A \rightarrow B : \{n\}_{w_{AB}}$
1. $B \rightarrow A : \{n+1\}_{w_{AB}}$

Aims to **authenticate** principal B from A 's viewpoint.

Example of guessing attack

Handshake Protocol

0. $A \rightarrow B$: $\{n\}_{w_{AB}}$ as m_1
1. $B \rightarrow A$: $\{n+1\}_{w_{AB}}$ as m_2

Aims to **authenticate** principal B from A 's viewpoint.

A **passive attacker** (pure eavesdropper) may recover shared password w_{AB} by guessing x such that

$$\text{dec}(m_1, x) + 1 \stackrel{?}{=} \text{dec}(m_2, x)$$

In practice, password-based encryption impl. by **keyed permutations**.

Frames

Terms (messages) are organized into frames:

$$\varphi_1 = \nu k_1, k_2, k_3, k_4. \{x_1 = \{k_1\}_{k_2}, x_2 = \{k_4\}_{k_3}, x_3 = k_3\}$$

Frames

Terms (messages) are organized into frames:

$$\varphi_1 = \nu k_1, k_2, k_3, k_4. \{x_1 = \{k_1\}_{k_2}, x_2 = \{k_4\}_{k_3}, x_3 = k_3\}$$

Definition (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

For the equational theory $\text{dec}(\{x\}_y, y) = x$,

- 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 $\text{var}(M, N) \subseteq \text{dom}(\varphi_1)$ and $\text{names}(M, N) \cap \text{names}(\varphi_1, \varphi_2) = \emptyset$,

$$M\varphi_1 =_E N\varphi_1 \quad \text{iff} \quad M\varphi_2 =_E N\varphi_2$$

Decision **procedures** exist for many theories E , see e.g. [Abadi, Cortier–ICALP'04 & CSFW'05].

Examples: keyed permutations

Consider the equational theory E :

$$\text{dec}(\{x\}_y, y) = x \quad \{ \text{dec}(x, y) \}_y = x$$

We have:

$$\nu k. \{x = \{0\}_k\} \text{ ? } \nu k. \{x = \{1\}_k\}$$

$$\{x = \{0\}_{c_0}\} \text{ ? } \{x = \{0\}_{c_1}\}$$

$$\nu n. \{x = \{n\}_{c_0}\} \text{ ? } \nu n. \{x = \{n\}_{c_1}\}$$

$$\nu n. \{x = \{n\}_{c_0}, y = \{s(n)\}_{c_0}\} \text{ ? } \nu n. \{x = \{n\}_{c_1}, y = \{s(n)\}_{c_1}\}$$

Examples: keyed permutations

Consider the equational theory E :

$$\text{dec}(\{x\}_y, y) = x \quad \{ \text{dec}(x, y) \}_y = x$$

We have:

$$\nu k. \{x = \{0\}_k\} \approx_E \nu k. \{x = \{1\}_k\}$$

$$\{x = \{0\}_{c_0}\} \not\approx_E \{x = \{0\}_{c_1}\}$$

$$\nu n. \{x = \{n\}_{c_0}\} \approx_E \nu n. \{x = \{n\}_{c_1}\}$$

$$\nu n. \{x = \{n\}_{c_0}, y = \{s(n)\}_{c_0}\} \not\approx_E \nu n. \{x = \{n\}_{c_1}, y = \{s(n)\}_{c_1}\}$$

(The last case is the attack on the [Handshake Protocol](#).)

Concrete implementation

- Complexity parameter η
- Assume a **PTIME** implementation for each function symbol, and **PPTIME** random generators for names.
- Terms t mapped to (distributions over) bit-strings $\llbracket t \rrbracket_\eta$
- We may restrict terms to **well-sorted** ones

Definition (indistinguishability)

$\llbracket \varphi_1 \rrbracket \approx \llbracket \varphi_2 \rrbracket$ if $\text{Adv}^{\text{IND}}(\mathcal{A}, \llbracket \varphi_1 \rrbracket_\eta, \llbracket \varphi_2 \rrbracket_\eta)(\eta) =$

$$\mathbb{P}[\phi_1 \leftarrow \llbracket \varphi_1 \rrbracket_\eta; \mathcal{A}(\eta, \phi_1) = 1] - \mathbb{P}[\phi_2 \leftarrow \llbracket \varphi_2 \rrbracket_\eta; \mathcal{A}(\eta, \phi_2) = 1]$$

is a **negligible** function of η .

Sorts, symbols and equational theory of interest

$\tau ::=$

	<i>SKey</i>	symmetric keys
	<i>EKey</i>	(public) encryption keys
	<i>DKey</i>	(private) decryption keys
	<i>Data</i>	passwords and other data
	<i>Coins</i>	coins for encryption
	<i>Pair</i> $[\tau_1, \tau_2]$	pairs of messages
	<i>SCipher</i> $[\tau]$	symmetric encryptions
	<i>ACipher</i> $[\tau]$	asymmetric encryptions

Sorts, symbols and equational theory of interest

$$\begin{array}{l}
 \text{enc}_{\tau} : \tau \times \text{Data} \rightarrow \tau \quad \tau \neq \text{Pair}[\tau_1, \tau_2] \\
 \text{dec}_{\tau} : \tau \times \text{Data} \rightarrow \tau \quad \text{"} \\
 \\
 \text{penc}_{\tau} : \tau \times \text{EKey} \times \text{Coins} \rightarrow \text{ACipher}[\tau] \\
 \text{pdec}_{\tau} : \text{ACipher}[\tau] \times \text{DKey} \rightarrow \tau \\
 \text{pub} : \text{DKey} \rightarrow \text{EKey} \\
 \text{pdec_success}_{\tau} : \text{ACipher}[\tau] \times \text{DKey} \rightarrow \text{Data} \\
 \\
 \text{senc}_{\tau} : \tau \times \text{SKey} \times \text{Coins} \rightarrow \text{SCipher}[\tau] \\
 \text{sdec}_{\tau} : \text{SCipher}[\tau] \times \text{SKey} \rightarrow \tau \\
 \text{sdec_success}_{\tau} : \text{SCipher}[\tau] \times \text{SKey} \rightarrow \text{Data} \\
 \\
 \text{pair}_{\tau_1, \tau_2} : \tau_1 \times \tau_2 \rightarrow \text{Pair}[\tau_1, \tau_2] \\
 \text{fst}_{\tau_1, \tau_2}, \text{snd}_{\tau_1, \tau_2} : \text{Pair}[\tau_1, \tau_2] \rightarrow \tau_2 \\
 \\
 0, 1, w, c_0, c_1 \dots : \text{Data}
 \end{array}$$

Sorts, symbols and equational theory of interest

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

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

$$\text{pdec}(\text{penc}(x, \text{pub}(y), z), y) = x$$

$$\text{pdec_success}(\text{penc}(x, \text{pub}(y), z), y) = 1$$

$$\text{sdec}(\text{senc}(x, y, z), y) = x$$

$$\text{sdec_success}(\text{senc}(x, y, z), y) = 1$$

$$\text{fst}(\text{pair}(x, y)) = x$$

$$\text{snd}(\text{pair}(x, y)) = y$$

$$\text{pair}(\text{fst}(x), \text{snd}(x)) = x$$

(sorts omitted)

Examples

Let $\{x\}_w = \text{enc}(x, w)$, $\{x\}_k^r = \text{senc}(x, k, r)$, and $\{x\}_{pk}^r = \text{penc}(x, pk, r)$.

$$\{x = \{0\}_k^{r_1}, y = \{0\}_k^{r_2}\} \approx_E \{x = \{1\}_{k_1}^{r_1}, y = \{0\}_{k_2}^{r_2}\}$$

$$\{x = \{\{n\}_w\}_w, y = \{m\}_w\} \approx_E \{x = a_1, y = a_2\}$$

$$\{x = \{\{0\}_{pk}^{r_1}\}_w, y = \{0\}_{pk}^{r_2}\} \approx_E \{x = a_1, y = a_2\}$$

$$\{x = \{\{0\}_{pk}^{r_1}\}_w, y = \{0\}_{pk}^{r_1}\} \approx_E \{x = \{a_1\}_w, y = a_1\}$$

$$\{x = \{\{n\}_k^{r_1}\}_w, y = k\} \not\approx_E \{x = a_1, y = k\}$$

(restrictions $\nu n, \nu a, \nu k, \nu r \dots$ omitted.)

Computational soundness of \approx_E

A reduced frame φ is **well-formed** if

- it contains no destructors ($\text{dec}, \text{pdec}, \dots$),
- encryption keys are either names k , of the form $\text{pub}(k)$, or constants of sort *Data*,
- coins are “fresh” names,
- φ has **no key encryption cycles** e.g. $(\{k\}_{\text{pub}(sk)}, \{sk\}_k)$,
- for every subterm $\text{enc}(T, k)$, T contains no constants $w, c_0, \dots, 0, 1$.

Theorem

*In any **secure** implementation, for every well-formed frames φ_1 and φ_2 , $\varphi_1 \approx_E \varphi_2$ implies $\llbracket \varphi_1 \rrbracket \approx \llbracket \varphi_2 \rrbracket$.*

Application to guessing attacks

Corollary

In any *secure* implementation, for every well-formed frame φ , $\varphi\{w \mapsto c_0\} \approx_E \varphi\{w \mapsto c_1\}$ implies that w is *computationally hidden* in φ : for every PTIME sequences κ_0 and κ_1 ,

$$\llbracket \varphi \rrbracket_{w \mapsto \kappa_0} \approx \llbracket \varphi \rrbracket_{w \mapsto \kappa_1}$$

Generalizes to *multiple passwords*.

Example

Encrypted Key Exchange (EKE) protocol

0. $A \rightarrow B$: $\{pk\}_{w_{AB}}$
1. $B \rightarrow A$: $\{\{k_1\}_{pk}^{r_1}\}_{w_{AB}}$
2. $A \rightarrow B$: $\{k_A\}_{k_1}^{r_2}$
3. $B \rightarrow A$: $\{\text{pair}(k_A, k_B)\}_{k_1}^{r_3}$
4. $A \rightarrow B$: $\{k_B\}_{k_1}^{r_4}$

Let $\varphi = \nu pk, sk, k_1, k_A, k_B, r_1 \dots r_4. \{ x_1 = \{pk\}_{w_{AB}}, x_2 = \{\{k_1\}_{pk}^{r_1}\}_{w_{AB}}, x_3 = \{k_A\}_{k_1}^{r_2}, x_4 = \{\text{pair}(k_A, k_B)\}_{k_1}^{r_3}, x_5 = \{k_B\}_{k_1}^{r_4} \}$.

We have $\varphi\{w_{AB} \mapsto c_0\} \approx_E \varphi\{w_{AB} \mapsto c_1\}$.

Summary and further work

- Computational soundness of static equivalence for encryption(s) and passwords.
- Natural application to guessing attacks.
- May wish to add modular exponentiation.
- Towards computational justification of process equivalences ?

Thanks!

Secure implementation : symmetric encryption

Let $\tau \in T_{\text{senc}}$, and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be 2-stage adversary.

- $k \xleftarrow{R} \mathcal{K}^s(\eta)$;
- A_1 is provided access to an oracle $\mathcal{E}^s(\cdot, k)$;
- then A_1 outputs a challenge message $m^* \in \llbracket \tau \rrbracket_\eta$ together with some state information st ;
- a bit $b \xleftarrow{R} \{0, 1\}$ is selected at random; if $b = 0$, we let $c \xleftarrow{R} \text{"SCipher"} \parallel \tau \parallel \mathcal{E}^s(m^*, k)$; otherwise, we let $c \xleftarrow{R} \llbracket \text{SCipher}[\tau] \rrbracket_\eta$;
- A_2 is given c and st , and outputs a bit b' .
- A is successful if $b' = b$.

$$\text{Adv}_{\Pi^s, A}^\tau(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$$

Secure implementation : asymmetric encryption

Let $\tau \in T_{\text{penc}}$, and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be 2-stage adversary.

- $(pk, sk) \xleftarrow{R} \mathcal{K}^a(\eta)$;
- \mathcal{A}_1 is given pk ;
- then \mathcal{A}_1 outputs a challenge message $m^* \in \llbracket \tau \rrbracket_\eta$ together with some state information st ;
- a bit $b \xleftarrow{R} \{0, 1\}$ is selected at random; if $b = 0$, we let $c \xleftarrow{R} \text{"ACipher"} \parallel_\tau \mathcal{E}^a(m^*, pk)$; otherwise, we let $c \xleftarrow{R} \llbracket \text{ACipher}[\tau] \rrbracket_\eta$;
- \mathcal{A}_2 is given c and st , and outputs a bit b' .
- \mathcal{A} is successful if $b' = b$.

$$\text{Adv}_{\Pi^a, \mathcal{A}}^\tau(\eta) = \Pr[\mathcal{A} \text{ is successful}] - \frac{1}{2}$$

Secure implementation : password encryption (1)

Let $\tau \in T_{\text{enc}}$, and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be 2-stage adversary.

- $k \xleftarrow{R} \mathcal{K}(\eta)$;
- A_1 is provided access to an oracle $\mathcal{E}(\cdot, k)$;
- then A_1 outputs a challenge message $m^* \in \llbracket \tau \rrbracket_\eta$ together with some state information st ;
- a bit $b \xleftarrow{R} \{0, 1\}$ is selected at random; if $b = 0$, we let $c \xleftarrow{R} \mathcal{E}(m^*, k)$; otherwise, we let $c \xleftarrow{R} \llbracket \tau \rrbracket_\eta$;
- A_2 is given c and st , and outputs a bit b' .
- A is successful if $b' = b$, and the challenge message m^* is different from all the messages m submitted by A to the encryption oracle..

$$\text{Adv}_{\text{RoR}, \Pi, \mathcal{A}}^\tau(\eta) = \Pr[A \text{ is successful}] - \frac{1}{2}$$

Secure implementation : password encryption (2)

Let $\tau \in T_{\text{enc}}$, and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be 2-stage adversary.

- \mathcal{A}_1 outputs a key $k \in \{0, 1\}^{\alpha_1(\eta)}$ and some state information st ;
- a bit $b \xleftarrow{R} \{0, 1\}$ is selected at random; if $b = 0$, we let $m \xleftarrow{R} \llbracket \tau \rrbracket_\eta$ and $c = \mathcal{E}(m, k)$; otherwise, we let $c \xleftarrow{R} \llbracket \tau \rrbracket_\eta$;
- \mathcal{A}_2 is given c and st , and outputs a bit b' .
- \mathcal{A} is successful if $b' = b$.

$$\text{Adv}_{\text{Pwd}, \Pi, \mathcal{A}}^\tau(\eta) = \Pr[\mathcal{A} \text{ is successful}] - \frac{1}{2}.$$