A Machine-Checked Formalization of the Generic Model and the Random Oracle Model

Sabrina Tarento

joint work with Gilles Barthe and Jan Cederquist

INRIA Sophia Antipolis

Overview

- Coq
- Perfect Cryptography assumption
- ElGamal
- Generic Model
- Formalization of generic algorithm
- Results on the GM
- Random Oracle Model
- Formalization of interactive generic algorithm
- Conclusion

Coq

- developed at INRIA and at the University of Paris Sud

- System allowing the development and the checking of mathematical proofs in a higher order logic

- based on Calculus of Inductive Constructions

- Types of lists, rings, ...

- The objects of Coq are (dependently) typed functional programs/ proofs are objects

- Construction of interactive proofs, using tactics, backwards construction

Formalization of Mathematics in Coq

- **no quotients** $\Rightarrow$ use of the setoids

  setoid: a set provided with a relation of equivalence.

  $$mod := \forall q \in \mathbb{N}\ \forall a, b \in \mathbb{Z}\ \exists k \in \mathbb{Z} \mid a - b = k \times q$$

- finite sets and probabilities

- modular development of polynomials

  Poly: Ring $\rightarrow$ Var $\rightarrow$ Ring

  Lemma 1 (Schwartz) *Let* $p(x_1, \ldots, x_k)$ *be a polynomial in* k *variables, not identical to* 0*, with degree at most* d*, and the values chosen uniformly and independently in* $[0, q-1]$*. Then* $Pr[p(x_1, \ldots, x_k) = 0] \leq d/q$*.*

Perfect cryptography assumption

there is no way to obtain knowledge about the plaintext pertaining to a ciphertext without knowing the key.

Assumption taken e.g in:

- belief logics (M.Burrows, M.Abadi, and R.Needham)
- model checkers (G.Lowe)
- proof assistants (L.C. Paulson)

Generic Model

- introduced by Shoup in 1997 and extended by Schnorr and Jakobsson
- Focus on attacks that work for all groups
- attackers make group operations and tests of collisions in order to find information about secrets
- ideal model with some difficulties (the same as ROM), but useful to prove security and well adapted to Coq
- used for proving the security of ECDSA ...

Running Example: ElGamal

G cyclic group of prime order q with generator g

A chooses randomly $x \in \{0, \ldots, q-1\}$

G, g et g^x are public data

$$A \longrightarrow B : \; g^x$$

B wants to send the message m to A, so B chooses randomly $r \in \{0, \ldots, q-1\}$

$$B \longrightarrow A : \; (g^r, m \cdot (g^x)^r)$$

decryption: $(g^r)^{-x} \cdot m \cdot (g^x)^r = m$

What is a random value?

- secrets are random in $\mathbb{Z}_q$
- Rather than formalizing random elements, we introduce a type *Sec* of secrets, use an interpretation function $f : \mathit{Sec} \to \mathbb{Z}_q$ and treat input and output are polynomials in $\mathbb{Z}_q[\mathit{Sec}]$
- the probability space is: $\mathit{Sec} \to \mathbb{Z}_q$

Generic algorithm

a *generic algorithm* performs t generic steps

- $f_1, \ldots, f_{t'} \in G$ (inputs) $1 \leq t' < t$,
- $f_i = \prod_{j=1}^{i-1} f_j^{a_j}$ for $i = t' + 1, \ldots, t$ where $(a_1, \ldots, a_{i-1}) \in \mathbb{Z}_q^{i-1}$

modeled as a list:

$$\frac{}{\mathit{empty\_run} \in \mathit{Run}}$$

$$\frac{R \in \mathit{Run} \quad e \in (\mathit{list}\ \mathbb{Z}_q)}{\mathit{step}(R, e) \in \mathit{Run}}$$

Collisions

$$CO_t := \{(j,k) | f_j = f_k, 1 \leq j < k \leq t\}$$

- only non trivial collision reveals information about the secrets

  example of ElGamal: $g, \; g^x, \; g^r, \; m_b \cdot (g^x)^r$

$$log_g f_i = a_{i,1} + a_{i,2}x + a_{i,3}r + a_{i,4}(log_g m_b + rx)$$

- finding informations about the secrets amounts solving

$$log_g f_i - log_g f_j = c_1 + c_2 x + c_3 r + c_4(log_g m_b + rx) = 0$$

Generic algorithm

$$GA = \{Sec : Set;\ run : Run;\ inp : (listT\ \mathbb{Z}_q[Sec]);\ condition\}$$

- Output: the list of polynomials in $\mathbb{Z}_q[Sec]$ resulting from multivariate exponentiations
- Concrete output: a list of elements in $\mathbb{Z}_q$ using the interpretation function $f : Sec \rightarrow \mathbb{Z}_q$ to the output
- Collisions: tests of non trivial equalities between the concrete outputs
- Condition: eliminate trivial equalities

Results

Lemma 2 $Pr(\mathcal{CO}_t \neq \emptyset) \leq \theta(\frac{d}{q} \cdot t^2)$.

Proof: use of Schwartz lemma.

Lemma 3 $Pr(SecFound\ x) \leq Pr(\mathcal{CO}_t \neq \emptyset) + Pr(guess\ x)$

Application to ElGamal

- **Generic DL-complexity lower bound:**
  inputs: $g,\ g^x$

$$Pr(\text{SecFound } x) \leq \theta(\frac{t^2}{q}) + \theta(\frac{1}{q})$$

- **Indistinguishability:**
  inputs: $g,\ g^x,\ g^r,\ m_b \cdot (g^x)^r,\ m_0,\ m_1$

$$Pr(\text{SecFound } b) \leq \theta(\frac{t^2}{q}) + \frac{1}{2}$$

Random Oracle Model

- group operations
- queries to the hash oracle H
- interactions with a decryption oracle

we find informations about the secrets by finding collisions or valid signed ciphertexts using interactions

- There exist signature and encryption schemes which are secure in the Random Oracle Model, but for which **any implementation** of the random oracle results in insecure schemes.(Canetti)

How do we formalize ideal hash function ?

- for communication with oracles: type *Val*

$$Val \perp\!\!\!\perp Sec$$

- interpretation function $rom : Val \rightarrow \mathbb{Z}_q$
- to formalize an hash function $H : \mathbb{Z}_q[Val]^3 - > Val$, we define a type $HashQuery := \mathbb{Z}_q[Val]^3 \times Val$

$$h : HashQuery := (a, b, d, c) \Rightarrow c = H(a, b, d)$$

Interactive generic algorithm

$$\frac{}{eRun : Run}$$

$$\frac{R : Run \quad e : list\ Val}{step(R, e) : Run}$$

$$\frac{R : Run \quad c : HashQuery}{hashstep(R, c) : Run}$$

$$\frac{R : Run \quad cip : \mathbb{Z}_q^4}{decstep(R, cip) : Run}$$

$$IGA = \{Sec, Val : Set;\ run : Run;\ inp : (listT\ (listT\ Val))\}$$

What to prove ?

Let a generic interactive algorithm be given g, g^x, m_0, m_1, cip_b and oracles for H and for decryption

Lemma 4 $Pr(SecFound\ b) \leq \theta(\frac{t^2}{q}) + \frac{1}{2}$

Conclusion

More future work

- Reason about attacks
- Can we extend Paulson's model with ideas from GM and ROM?
- Feedbacks welcome