Synthesis of OWF-based Encryption Schemes

Martin Gagné

Université Grenoble 1, CNRS, VERIMAG, FRANCE

joint work with

Gilles Bartes, Juan Manuel Crespo, Benjamen Grégoire, César Kunz, Yassine Lakhnech and Santiago Zanella-Béguelin

Submitted to POPL '13

Motivation

Use recent advances in automated proving to help discover and verify new constructions for encryption schemes

- build a synthesizer that outputs encryption scheme candidates
- use logic to filter out uninvertible candidates and discover decryption algorithm
- automatically prove IND-CPA security
- test for IND-CCA security

Synthesis of Encryption Schemes

Grammar for encryption algorithms:

$$e ::= r \mid 0 \mid m \mid f(e) \mid H(e) \mid e \oplus e \mid e \| e$$

Our encryption scheme synthesizer:

- generates all possible encryption algorithms requiring n commands
- uses symbolic logic to eliminate trivially insecure encryption scheme
- uses similar logic to synthesize decryption algorithm

Synthesis of Encryption Schemes

Deducibility logic rules:

$$\frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash e_1 \| e_2}\ \text{Conc} \qquad \frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash (e_1 \oplus e_2)\!\downarrow}\ \text{Xor} \qquad \frac{e \vdash e'}{e \vdash H(e')}\ \text{H}$$

$$\frac{e \vdash e_1 \| e_2}{e \vdash e_i}\ \text{Proj}_i \qquad \frac{e \vdash e'}{e \vdash f(e')}\ \text{f} \qquad \boxed{\frac{e \vdash f(e')}{e \vdash e'}\ \text{finv}}$$

- trivially insecure if you can deduce either r or m from ciphertext using non-boxed rules
- discover decryption algorithm by deducing m using all rules (including boxed)

IND-CPA Security

Proof search analyzes goals of the form (c, X, E).
Start with $(c, X, b = b')$ where c is expression for ciphertext, X is a list of all $H(e)$ in c
A goal is solvable if

- E is $b = b'$ and b does not appear in either c or X. The probability of E occurring is $1/2$.
- E of the form $e \in Q_H$ and e has a uniform random substring of length p. The probability of E occurring is bounded by $|Q_H|/2^p$
- E is of the form $e \in Q_H$, $f(r_1 \| \ldots \| r_n)$ is a substring of c with all r_i random, and a non-empty subset $R \subseteq \{r_1, \ldots, r_n\}$ can be deduced from e. The probability of E occurring is bounded by the probability of partially inverting f on R

IND-CPA Security

If goal (c, X, E) not solvable, modify goal using following rules:

- **Optimistic Sampling**: if r random, $r \oplus e$ sub-expression of c and r never used elsewhere, replace all instances of $r \oplus e$ by r' random
- **Permutation**: if r random, $x := f(r)$ and r never used again, replace by $x := r'$ for r' random
- **Failure Event**: find sub-expression $H(e)$ in c, set $c' = c\{r \ / \ H(e)\}$ and $X' = X - H(e)$, and solve goals (c', X', E) and $(c', X', e \in Q_H)$

* **Eager Sampling**: remove $H(e)$ from code of encryption algorithm if $H(e)$ does not appear in c

If rule * is not used, we can use EasyCrypt [BGHZ11] to produce proof with exact security bounds.

Chosen-Ciphertext Security

So far, no strategy to automatically generate sequence of games.
We instead prove a general criterion for plaintext awareness

$$H_0(t_0) \qquad H_1(t_1) \oplus t_0 \qquad \dots \qquad H_n(t_n) \oplus t_{n-1}$$

$$H_0(t_0) \text{ checkbits} \qquad t_n \vdash r \text{ or } G(r) \qquad t_n \| r \vdash m$$

Chosen-Ciphertext Security

So far, no strategy to automatically generate sequence of games.
We instead prove a general criterion for plaintext awareness

$$H_0(t_0) \qquad H_1(t_1) \oplus t_0 \qquad \dots \qquad H_n(t_n) \oplus t_{n-1}$$

$$H_0(t_0) \text{ checkbits} \qquad t_n \;\vdash\; r \text{ or } G(r) \qquad t_n \| r \;\vdash\; m$$

Limitations

- not tight
- unlikely to ever get general enough
- cannot work for IND-CCA schemes that are not plaintext aware

Experimental Results

- Our synthesizer can generate more than 100,000 candidate encryption schemes in a few hours
- Close to 3,000 IND-CPA schemes, close to 2,000 IND-CCA
- all the filters, IND-CPA proof and IND-CCA test take less than 10 minutes for all candidates

Experimental Results

- Our synthesizer can generate more than 100,000 candidate encryption schemes in a few hours
- Close to 3,000 IND-CPA schemes, close to 2,000 IND-CCA
- all the filters, IND-CPA proof and IND-CCA test take less than 10 minutes for all candidates

Size	Unfiltered	Filtered	CPA (PA)	Redundant CPA (PA)
2	15	1	1 (0)	7 (4)
3	211	17	16 (0)	122 (112)
4	22,856	1,818	1,178 (711)	15,606 (12,682)
5	85,910	2,203	1,653 (1,154)	24,305 (19,996)

Future Work

For current grammar:

- Further optimize synthesizer to increase number of candidates
- Prove (in)completeness of automatic semantic security prover
- Attempt to prove IND-CCA security directly with sequence of games

Future Work

For current grammar:

- Further optimize synthesizer to increase number of candidates
- Prove (in)completeness of automatic semantic security prover
- Attempt to prove IND-CCA security directly with sequence of games

Longer term:

- Use similar technique to generate schemes for larger set of complexity assumptions (Diffie-Hellman, lattices, etc)
- Develop new methods for proving security of encryption schemes with more complex security games (IBE, ABE, etc)
- Synthesis of signature, symmetric encryption, etc...