Two views on cryptographic reductions

Closing the gap between cryptography and program analysis

Sven Laur*
slaur@tcs.hut.fi

Helsinki University of Technology

Fools rush in where angels fear to tread.

... Although a construction of complex protocols is clearly an engineering task, security analysis of such protocols often resembles to witchcraft; sometimes the results are clearly wrong or too vague. Hence, many famous cryptographers have tried to come up with rigorous methods. Nevertheless, we still do not have formal machine verifiable methodology that is common in other engineering disciplines.

The Root of All Evil

What is conditional probability?

Consider a simplistic algorithm

Obfuscate(x) :

if $x = 0$

then $y \leftarrow 1$

else $y \leftarrow \{0, 1\}$ `/* y is chosen randomly */`

return y

What is the probability that $x = 1$ if $y = 1$?

(a) $\Pr[x = 1 | y = 1] = 0$

(b) $\Pr[x = 1 | y = 1] = \frac{1}{3}$

(c) $\Pr[x = 1 | y = 1] = 1$

Bayes Rule revisited

$$\Pr[\mathcal{A}|\mathcal{B}] := \frac{\Pr[\mathcal{B} \wedge \mathcal{A}]}{\Pr[\mathcal{B}]} = \frac{\Pr[\mathcal{B}|\mathcal{A}] \cdot \Pr[\mathcal{A}]}{\sum_{\mathcal{A}} \Pr[\mathcal{B}|\mathcal{A}] \cdot \Pr[\mathcal{A}]}$$

As the conditional probability depends on $\Pr[\mathcal{A}]$, it is dangerous to directly reason about $\Pr[\mathcal{A}|\mathcal{B}]$:

- $\Pr[\mathcal{A}]$ can be unintentionally changed during the argumentation.
- At the end of security analysis we obtain unconditional probabilities.
  - At some point we must multiply the result with $\Pr[\mathcal{B}]$.
  - We can work directly with joint probabilities $\Pr[\mathcal{A} \wedge \mathcal{B}]$.

Challenger-Centric Approach

Challenger-centric security game

Let for any t-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

$$\boxed{\mathsf{Adv}^{\mathsf{IND-CPA}}(\mathcal{A}) = \Pr\left[\mathcal{G}^{\mathcal{A}}_{\mathsf{IND-CPA}} = 1\right] \leq \tfrac{1}{2} + \varepsilon}$$

where

$\mathcal{G}^{\mathcal{A}}_{\mathsf{IND-CPA}}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}$

$(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(\mathsf{pk})$

$i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$

if $\mathcal{A}_2(\sigma, c) = i$ **then return** 1

else return 0

A simple example

Bound the success of the following lets-try-again game $\mathcal{G}_\star$

$\mathcal{G}_\star^{\mathcal{B}}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\overline{m})$

$(b, m_0, m_1, \sigma) \leftarrow \mathcal{B}_1(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$

  if $\mathcal{B}_2(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

provided that $\mathsf{Adv}^{\mathsf{IND-CPA}}(\mathcal{A}) \leq \frac{1}{2} + \varepsilon$ for any t-time $\mathcal{A}$.

Security proof

The game $\mathcal{G}_\star$ has very different structure from $\mathcal{G}_{\mathsf{IND-CPA}}$. Still:

- $\Pr\left[\mathcal{B}_2(\sigma, c) = i | b = \perp\right] \leq \frac{1}{2} + \varepsilon$ or $(\mathcal{B}_1, \mathcal{B}_2)$ wins the IND-CPA game.
- Similarly, $\Pr\left[(b, m_1, m_2, \sigma) \leftarrow \mathcal{B}_1(\mathsf{pk}, \overline{c}) : b = \overline{m} | b \neq \perp\right] \leq \frac{1}{2} + \varepsilon$ or $\mathcal{B}_1$ with a trivial sampler $m_1 = 0, m_2 = 1, \sigma = \mathsf{pk}$ wins the IND-CPA game.

Putting all together, we get

$$\begin{aligned}
\Pr\left[\mathcal{G}_\star^{\mathcal{B}} = 1\right] \leq & \Pr\left[b = \perp\right] \cdot \Pr\left[\mathcal{B}_2(\sigma, c) = i | b = \perp\right] \\
& + \Pr\left[b \neq \perp\right] \cdot \Pr\left[\mathcal{B}_1(\mathsf{pk}, \overline{c}) = \overline{m} | b \neq \perp\right] \\
\leq & \Pr\left[b = \perp\right] \cdot (\tfrac{1}{2} + \varepsilon) + \Pr\left[b \neq \perp\right] \cdot (\tfrac{1}{2} + \varepsilon) \\
\leq & \tfrac{1}{2} + \varepsilon \ .
\end{aligned}$$

Second attempt: Challenger-centric reduction

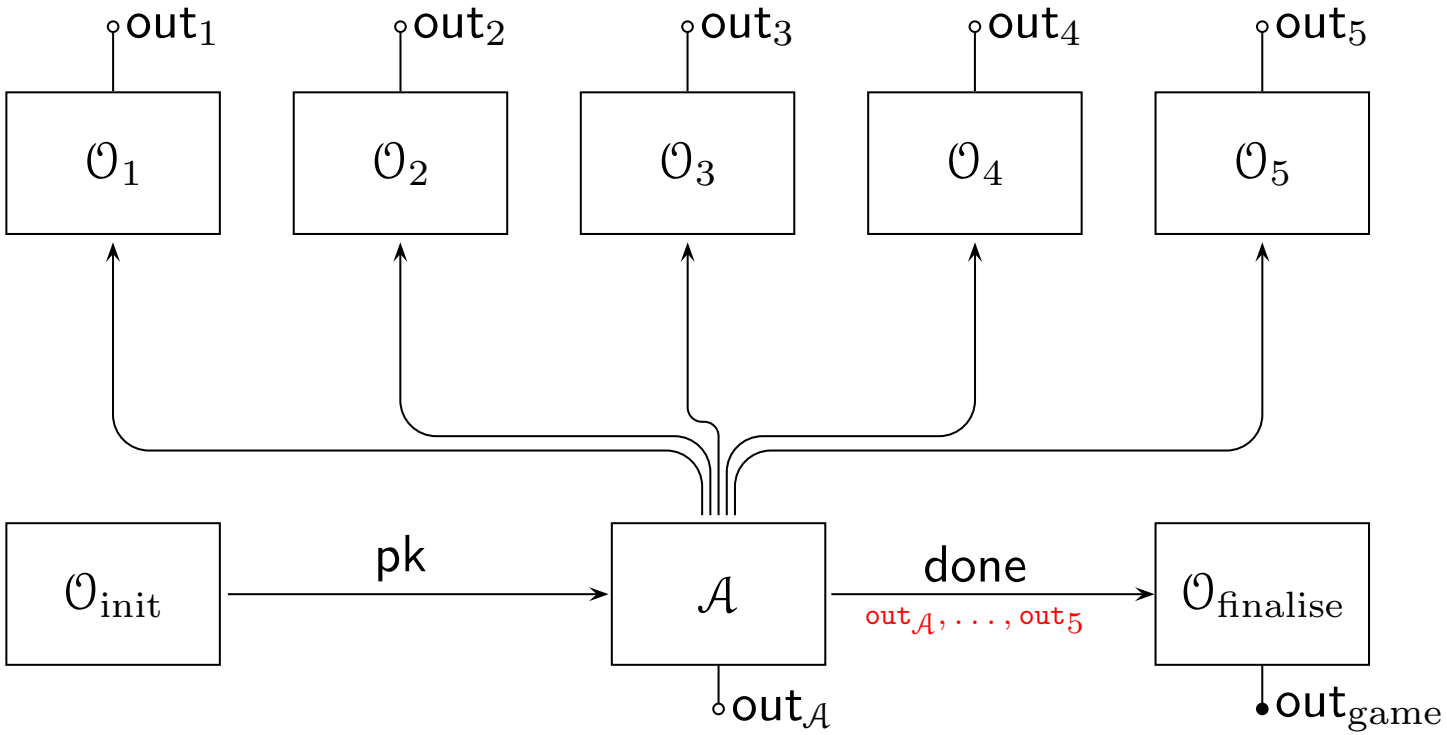
Main problem. The game $\mathcal{G}_\star$ has very different structure from $\mathcal{G}_{\mathsf{IND-CPA}}$.

- We must guess what attack path $\mathcal{B}_1$ chooses.
- For a fixed guess, set up the correct IND-CPA game.
- Finally, analyse the success probability without errors.

Main consequence.
Such reductions cannot be easily automatically generated nor verified.

Adversary-Centric Approach

Adversary-centric security game



Oracles $\mathcal{O}_{\mathrm{init}}, \mathcal{O}_1, \ldots, \mathcal{O}_n$ share a common memory, $\mathcal{O}_{\mathrm{init}}$ sets up global variables and pk. Outputs out_i are not observable by $\mathcal{A}$. Finally, $\mathcal{O}_{\mathrm{finalise}}$ computes the output $\mathsf{out}_{\mathrm{game}}$ given only $\mathsf{out}_{\mathcal{A}}, \mathsf{out}_1, \ldots, \mathsf{out}_n$.

Code transformation rules

Any security property can be defined as a transformation rule $(\mathcal{P}, \mathcal{T}, \mathcal{S})$:

- $\mathcal{P}$ temporal precondition that must be satisfied,
- $\mathcal{T}$ the actual syntactic code transformation schema,
- $\mathcal{S}$ achievable security guarantee.

For example, IND-CPA security is equivalent to a transformation rule

- $\mathcal{P}$: no information derived from sk are not released to $\mathcal{A}$.
- $\mathcal{T}$: replace $\mathsf{Enc}_{\mathsf{pk}}(x)$ by $\mathsf{Enc}_{\mathsf{pk}}(x_\circ)$ where $x_\circ \leftarrow \mathcal{D}$ and $\mathrm{supp}(\mathcal{D}) \subseteq \mathcal{M}$.
- $\mathcal{S}$: If the running-time of both games $\mathcal{G}_0$ and $\mathcal{G}_1$ is less than t, then

$$|\mathsf{Adv}^{\mathcal{G}_0}(\mathcal{A}) - \mathsf{Adv}^{\mathcal{G}_1}(\mathcal{A})| = \left|\Pr\left[\mathcal{G}_0^{\mathcal{A}} = 1\right] - \Pr\left[\mathcal{G}_1^{\mathcal{A}} = 1\right]\right| \leq \varepsilon \ .$$

Adversary-centric reduction I

Apply code transformation rules until you have reached trivial game.

$\mathcal{G}^{\mathcal{B}}_{\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\overline{m})$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$

  if $\mathcal{B}(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

$\xrightarrow[\varepsilon]{\mathsf{IND-CPA}}$

$\mathcal{G}^{\mathcal{B}}_{\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$

  if $\mathcal{B}(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

Adversary-centric reduction II

Apply code transformation rules until you have reached trivial game.

$\mathcal{G}^{\mathcal{B}}_{\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_i)$

  if $\mathcal{B}(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

$\xrightarrow[\varepsilon]{\mathsf{IND-CPA}}$

$\mathcal{G}^{\mathcal{B}}_{\star\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_0)$

  if $\mathcal{B}(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

Adversary-centric reduction III

Apply code transformation rules until you have reached trivial game.

$\mathcal{G}^{\mathcal{B}}_{\star\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{m} \leftarrow \{0,1\}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

if $b = \perp$ **then**

  $i \leftarrow \{0,1\}, c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_0)$

  if $\mathcal{B}(\sigma, c) = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

$\xrightarrow[0]{\text{DELAY}}$

$\mathcal{G}^{\mathcal{B}}_{\star\star\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

$\overline{m} \leftarrow \{0,1\}$

if $b = \perp$ **then**

  $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_0), b' \leftarrow \mathcal{B}(\sigma, c), i \leftarrow \{0,1\}$

  if $b' = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

Adversary-centric reduction IV

Apply code transformation rules until you have reached trivial game.

$\mathcal{G}^{\mathcal{B}}_{\star\star\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

$\overline{m} \leftarrow \{0,1\}$

if $b = \perp$ **then**

  $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_0), b' \leftarrow \mathcal{B}(\sigma, c), i \leftarrow \{0,1\}$

  if $b' = i$ **then return** 1 **else return** 0

else if $b = \overline{m}$ **then return** 1

else return 0

$\xrightarrow[\frac{1}{2}]{\mathsf{BAD}}$

$\mathcal{G}^{\mathcal{B}}_{\star\star\star\star\star}$:

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, \overline{c} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(0)$

(b, m_0, m_1, σ) get from $\mathcal{B}(\mathsf{pk}, \overline{c})$

$\overline{m} \leftarrow \{0,1\}$

if $b = \perp$ **then**

  $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_0), b' \leftarrow \mathcal{B}(\sigma, c), i \leftarrow \{0,1\}$

  if $b' = i$ **then return** 0 **else return** 0

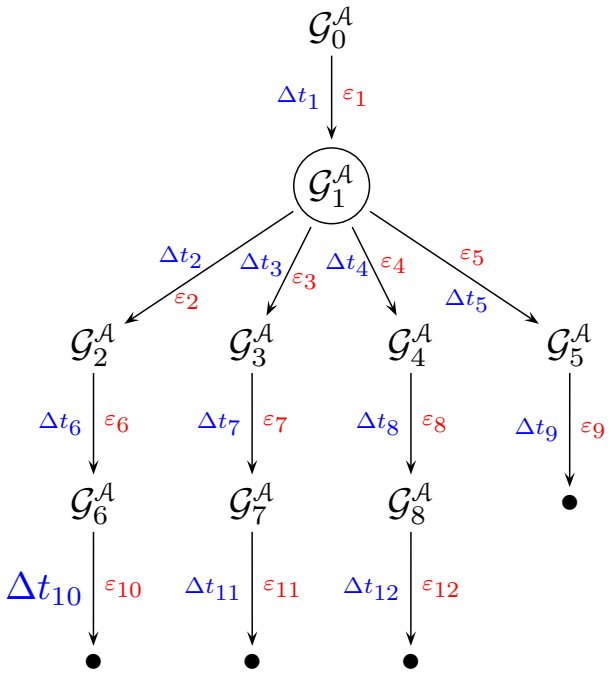
else if $b = \overline{m}$ **then return** 0

else return 0

$$\Pr\left[\mathcal{G}^{\mathcal{B}}_{\star} = 1\right] \leq \underbrace{\Pr\left[\mathcal{G}^{\mathcal{B}}_{\star\star\star\star\star} = 1\right]}_{0} + 2\varepsilon + \frac{1}{2} = 2\varepsilon + \frac{1}{2} \ .$$

Computer-Aided Security Proofs

General proof structure



Maximum working times t_i and probabilities p_i are computed bottom up

$$t_i = \min_{j \in \mathsf{childs}(i)} t_j - \Delta t_{ij} \quad \text{and} \quad p_i = \max_{j \in \mathsf{childs}(i)} p_j + \sum_{j \in \mathsf{childs}(i)} \varepsilon_{ij}$$

Where do we need program analysis?

To check the proof we must:

- Test that temporal preconditions $\mathcal{P}$ are satisfied.
- Evaluate BAD rule:
  - Find out which random variables can cause divergence in games.
  - Evaluate corresponding probability
- Evaluate DEAD CODE ELIMINATION rule

The code is multi-threaded if the original protocol is asynchronous.

Are such proofs sound and complete?

- All proofs are sound if all code transformation rules are valid.
- If some transformation rule is missing:
  - We can always bound the difference by hand.
  - Then formalise it as a rule and add it to system.
- Many proofs can be generated automatically using heuristics.

Questions! Answers?