Computer-aided cryptographic proofs

Pierre-Yves Strub

Modern cryptography

Shannon ’49

- Mathematical proof of security
- Perfect secrecy is impossible

Diffie & Hellman ’76

Goldwasser & Micali’82

Yao’82

- Computational security
- Asymptotic guarantees

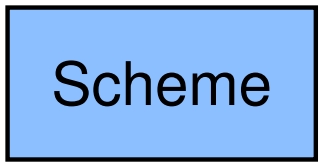
PPT adversary has negligible advantage

Bellare & Rogaway ’94

- Concrete bounds

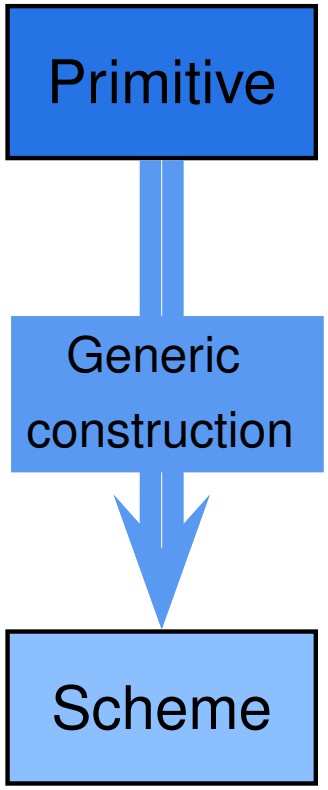
Aversary advantage to win in time t is $\leq p$

Reductionist proof

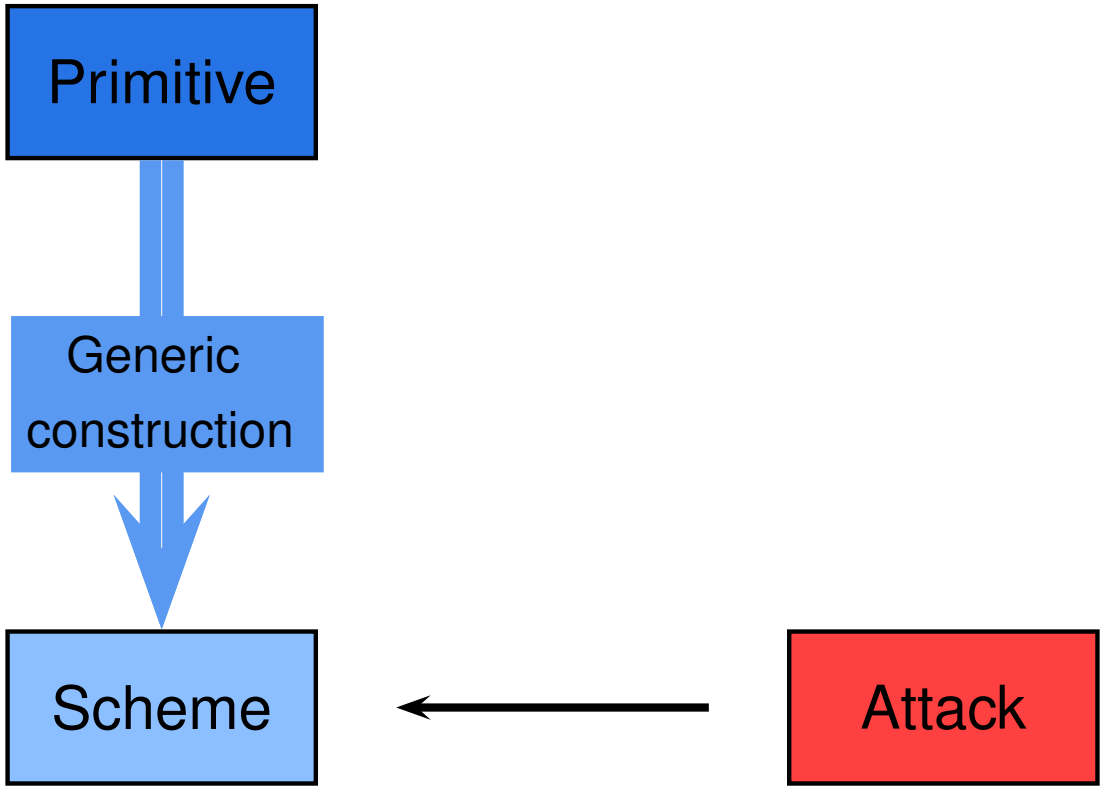


Reductionist proof

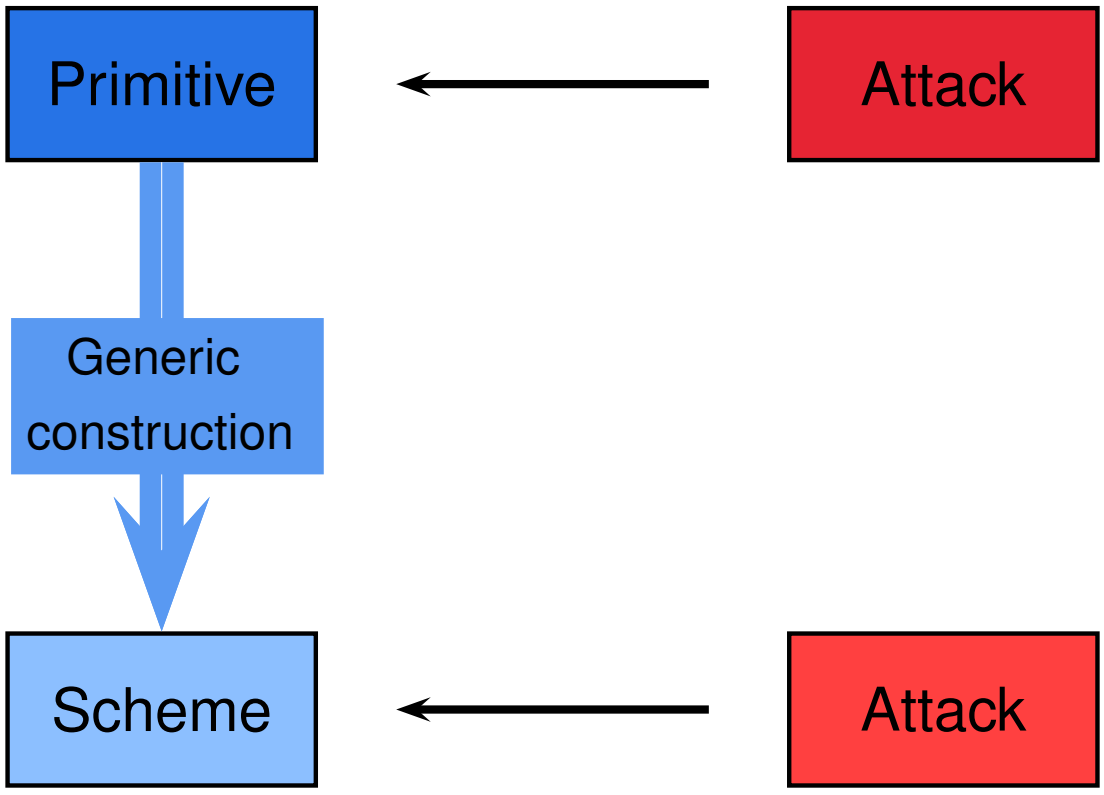
Reductionist proof



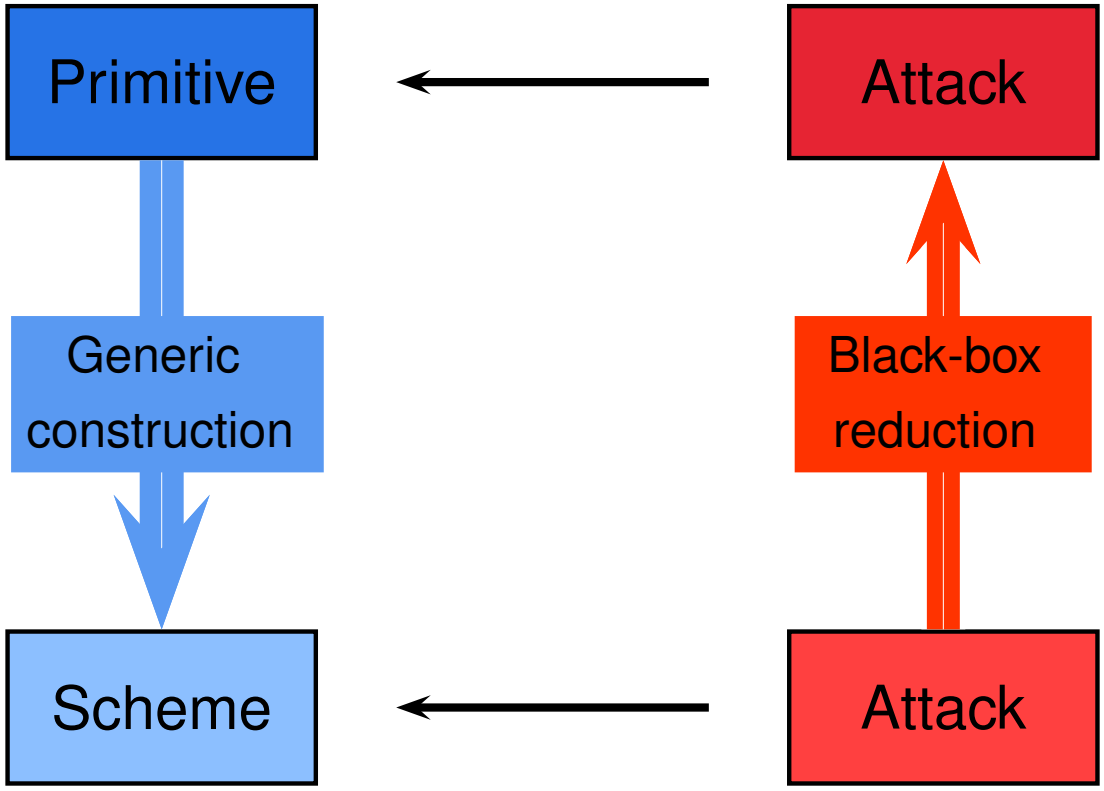
Reductionist proof



Reductionist proof



Reductionist proof



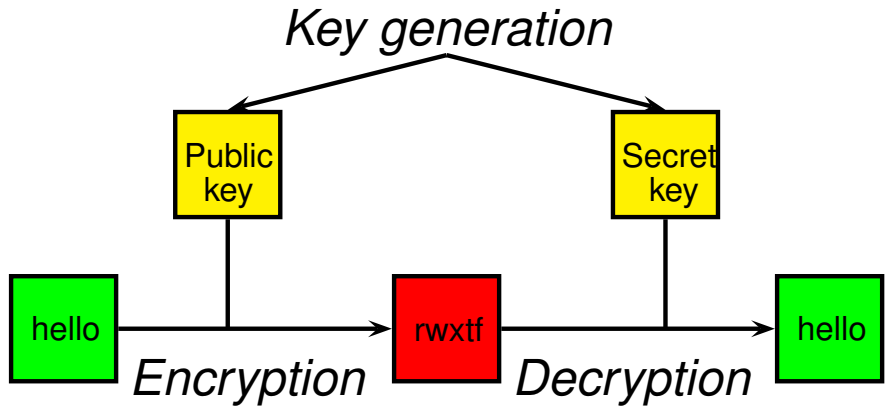
Public-key encryption

Algorithms $(\mathcal{K}, \mathcal{E}_{pk}, \mathcal{D}_{sk})$

- $\mathcal{E}$ probabilistic
- $\mathcal{D}$ deterministic and partial

If (sk, pk) is a valid key pair,

$$\mathcal{D}_{sk}(\mathcal{E}_{pk}(m)) = m$$

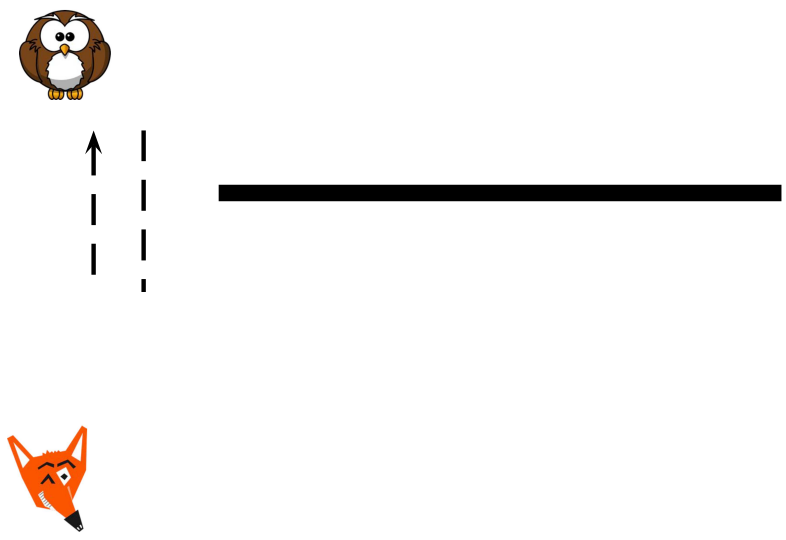


Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0, 1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$

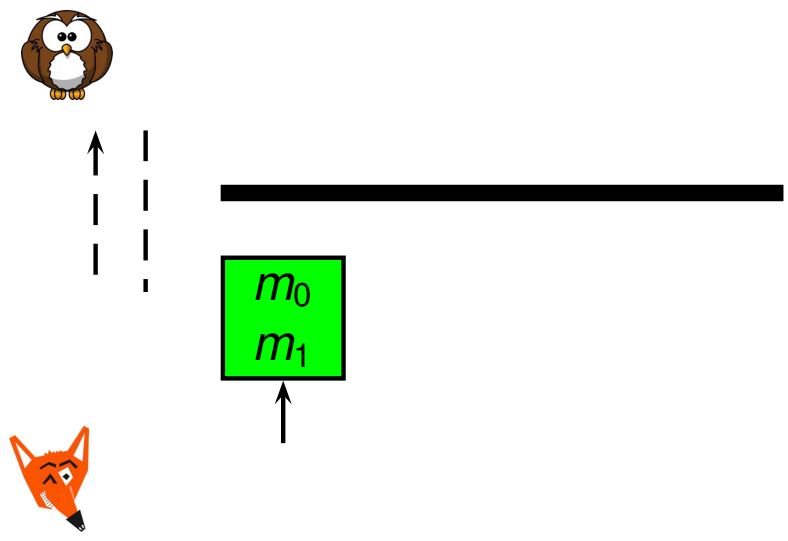
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0, 1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



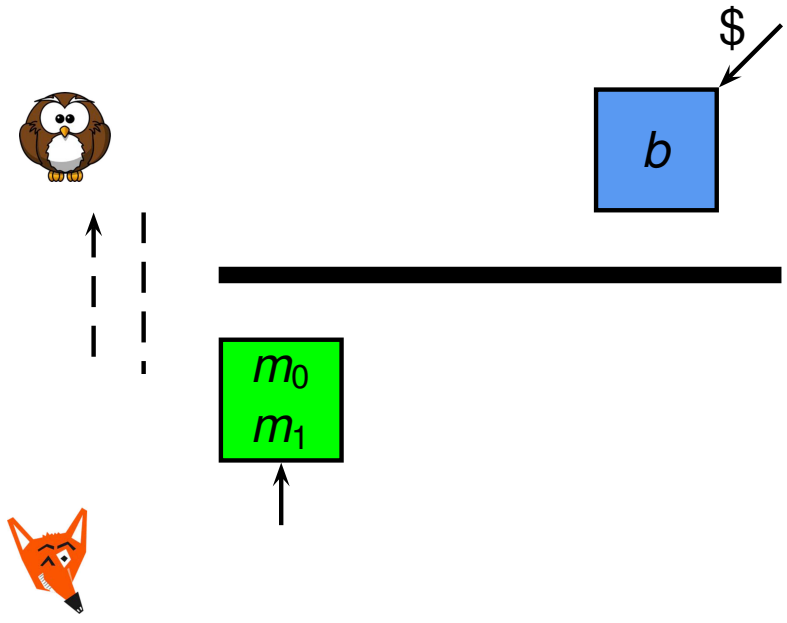
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0, 1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



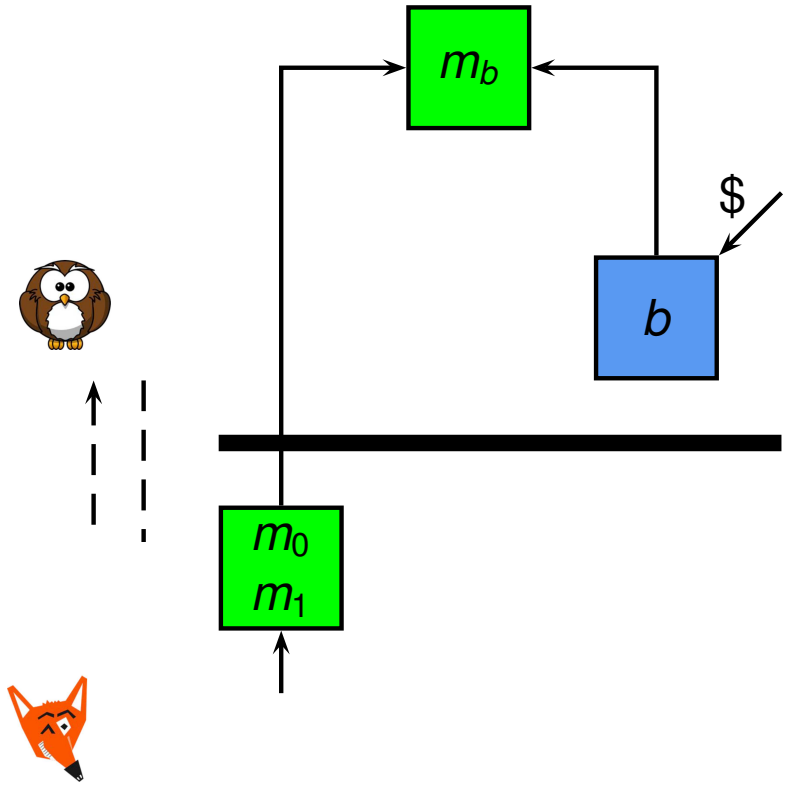
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



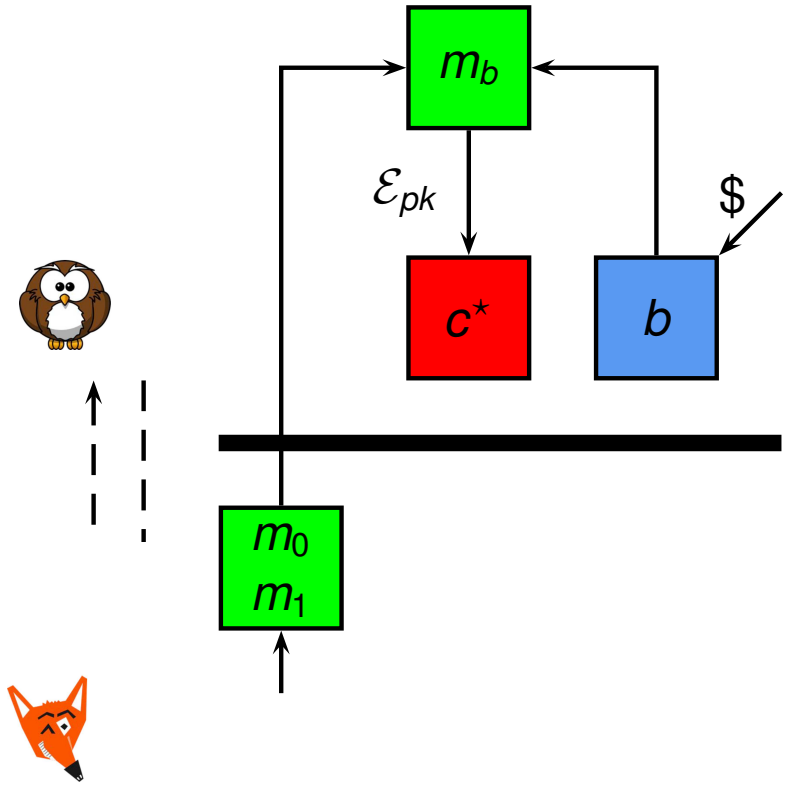
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0, 1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



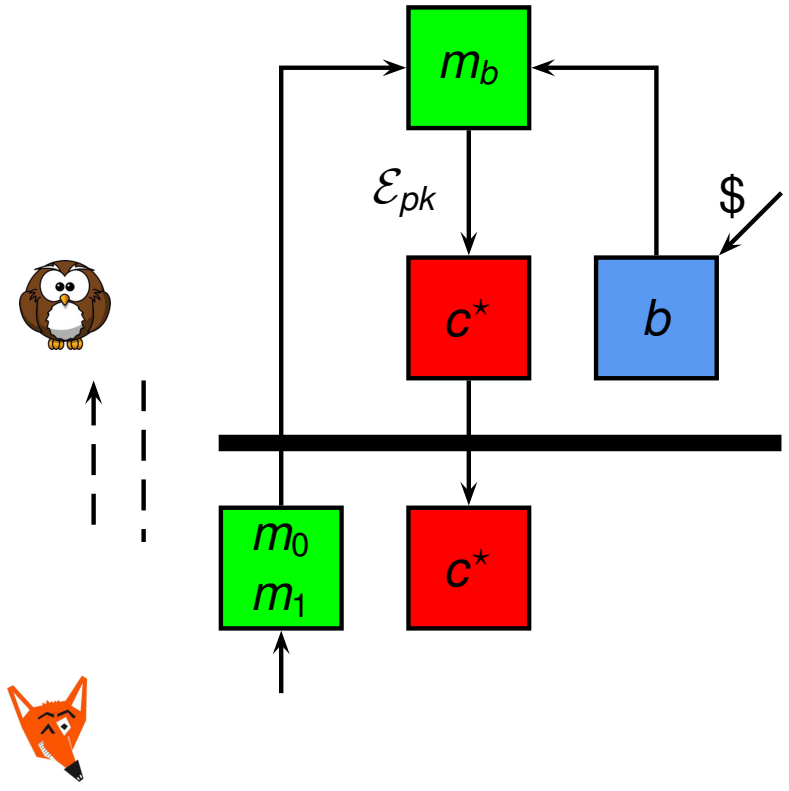
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



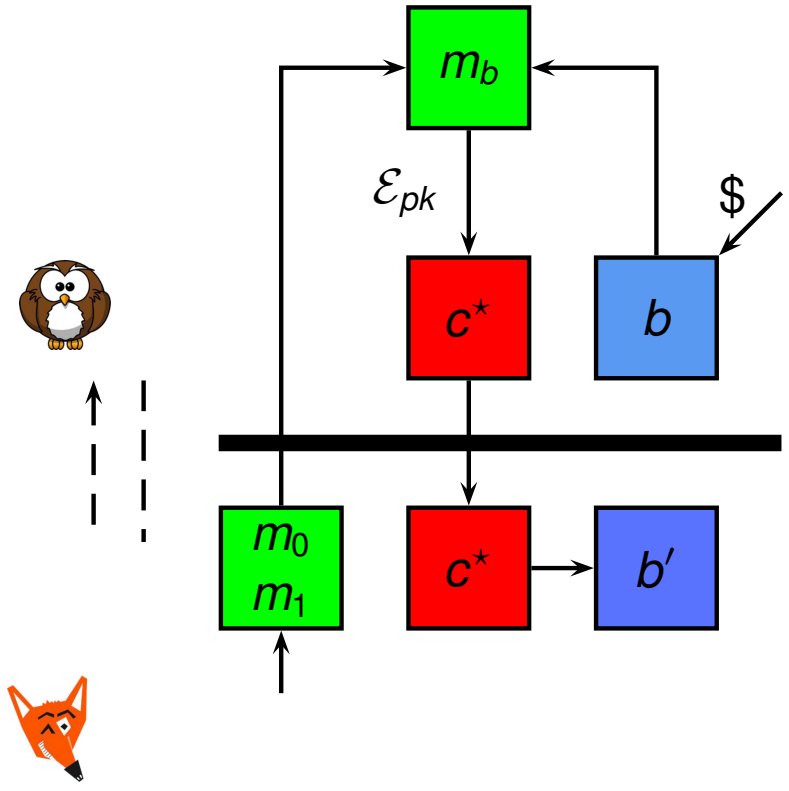
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



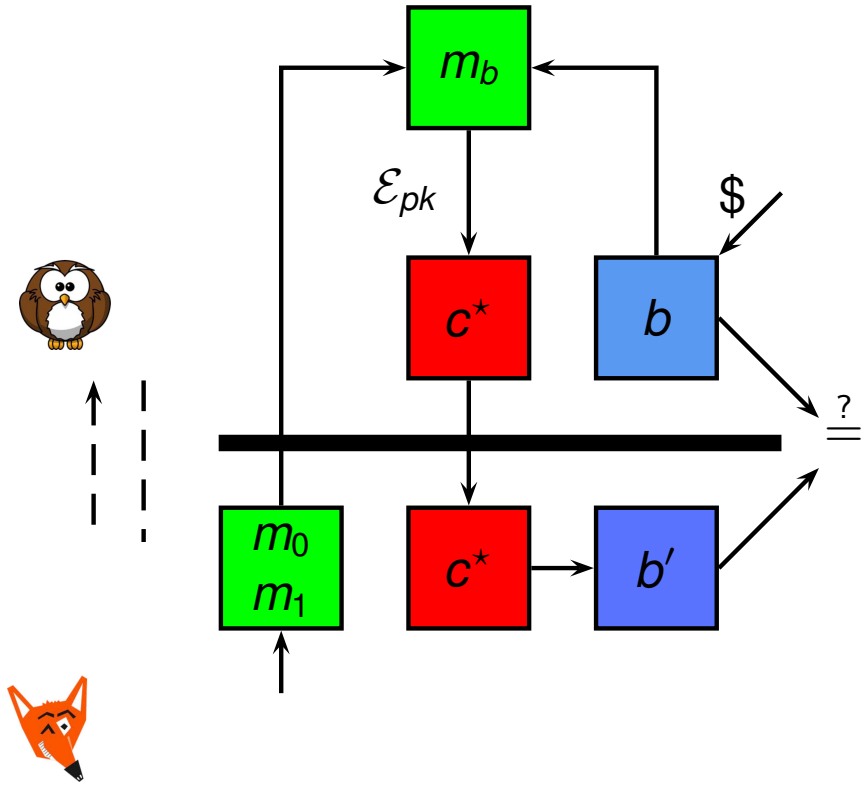
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



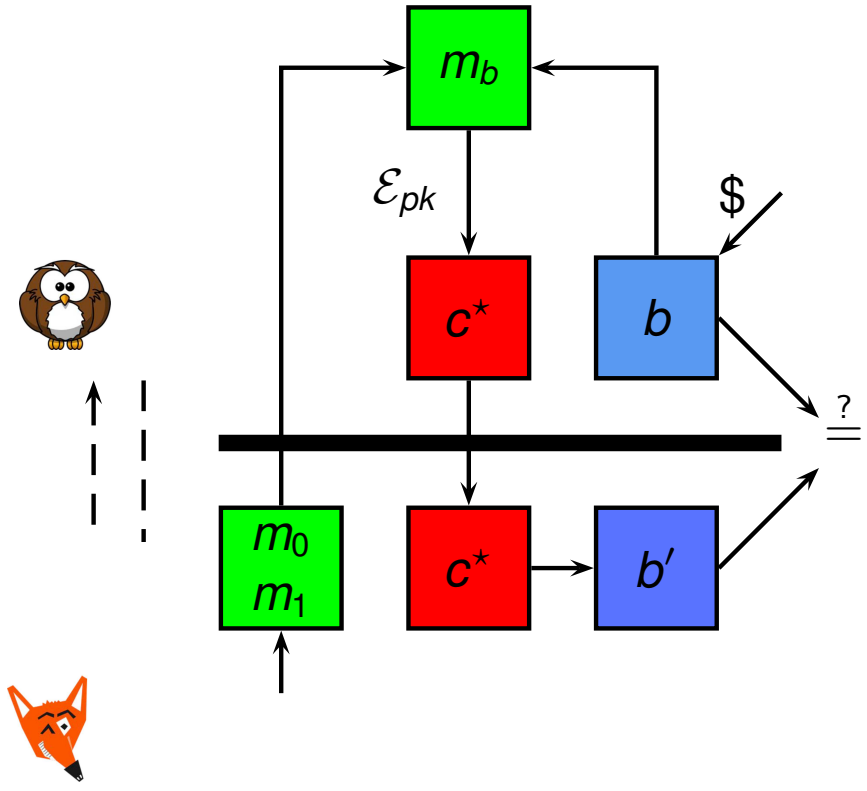
Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



Indistinguishability

Game IND-CPA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$



$$\Pr_{\text{IND-CPA}(\mathcal{A})}\left[b' = b\right] - \frac{1}{2} \quad \text{small}$$

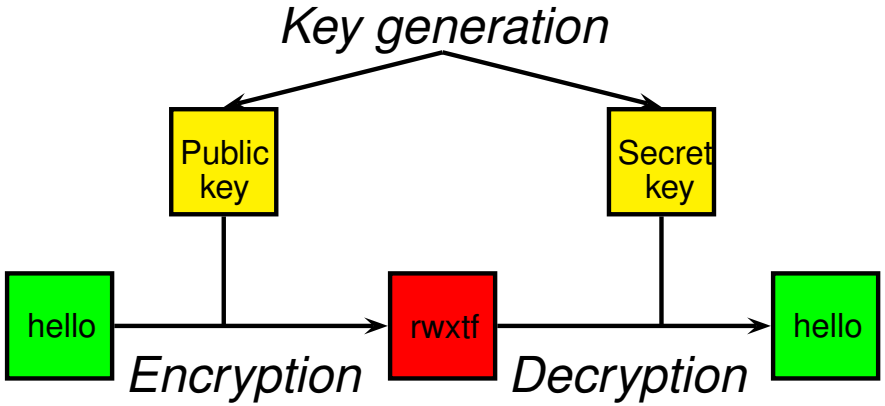
One-way trapdoor permutations

Algorithms $(\mathcal{K}, \mathsf{f}_{pk}, \mathsf{f}_{sk}^{-1})$

- f_{pk} and f_{sk}^{-1} deterministic

If (sk, pk) is a valid key pair,

$$\mathsf{f}_{sk}^{-1}(\mathsf{f}_{pk}(m)) = m$$

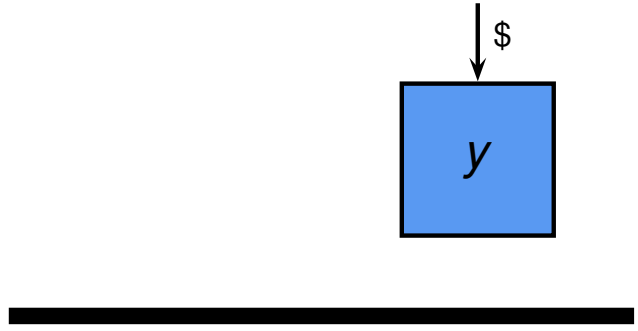


One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$

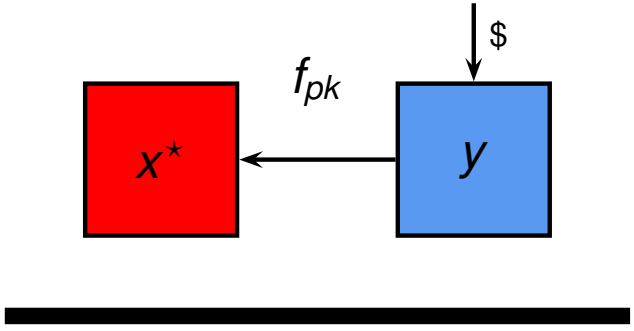
One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



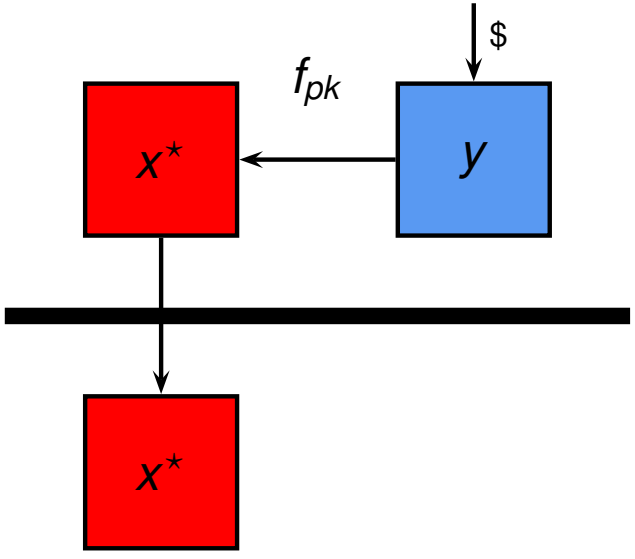
One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



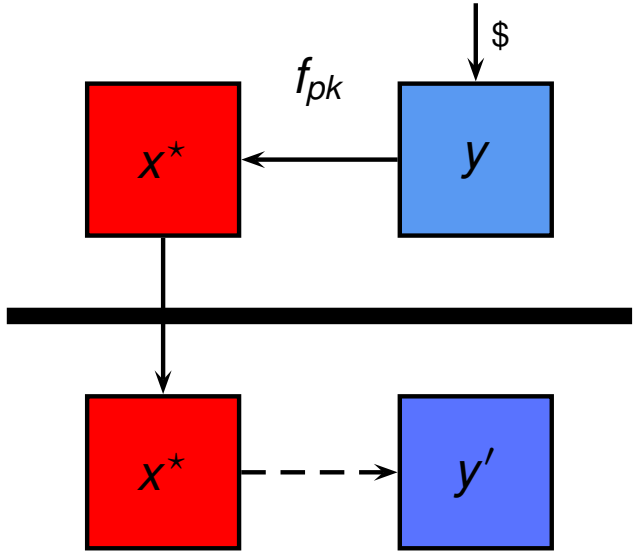
One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



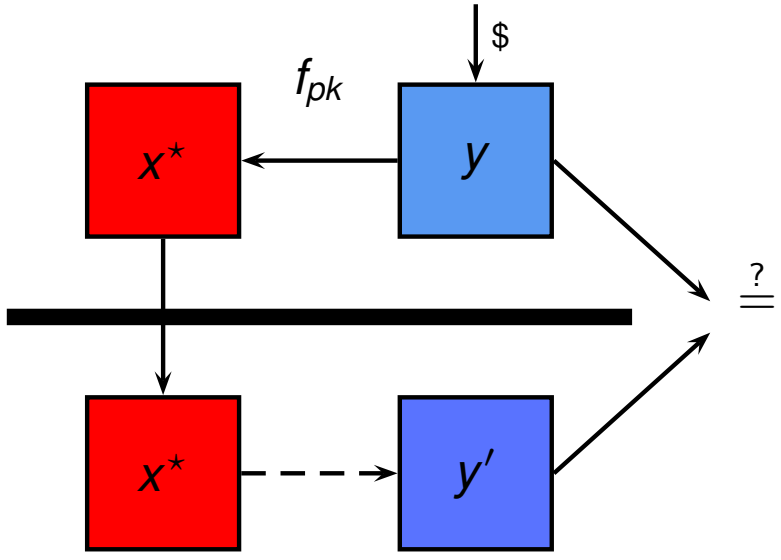
One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



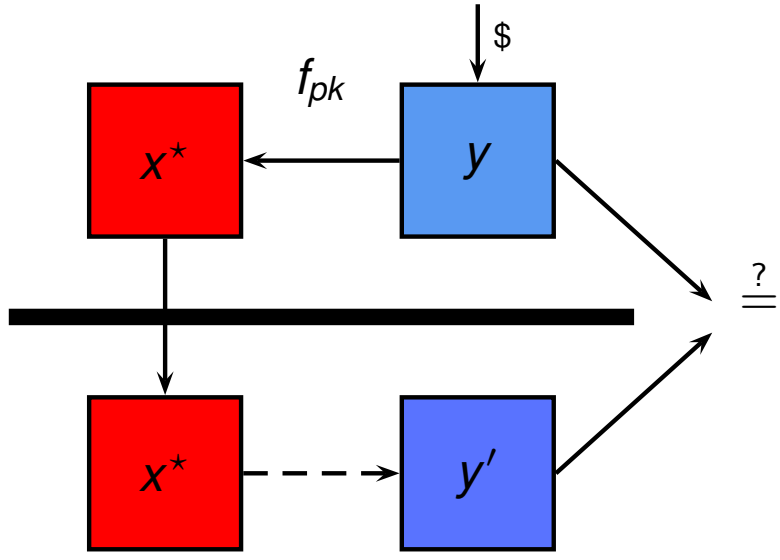
One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



One-way trapdoor permutations

Game $\mathrm{OW}(\mathcal{I})$
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^n;$
$x^\star \leftarrow \mathsf{f}_{pk}(y);$
$y' \leftarrow \mathcal{I}(x^\star);$
return $(y' = y)$



$$\Pr_{\mathrm{OW}(\mathcal{I})}\left[y' = y\right] \quad \text{small}$$

Random oracles

Oracle $H(x)$:
if $x \notin L$ then
  $r \xleftarrow{\$} \{0,1\}^k$;
  $L \leftarrow (x, r) :: L$;
return $L[x]$;

- Idealized model of hash function
- Allows practical schemes
- Not realizable

Example: Bellare and Rogaway 1993 encryption

Game IND-CPA($\mathcal{A}$) :
$(sk, pk) \leftarrow \mathcal{K}(\ )$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$

Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \leftarrow H(r) \oplus m$;
$y \leftarrow f_{pk}(r) \,\|\, s$;
return y

For every IND-CPA adversary $\mathcal{A}$, there exists an inverter $\mathcal{I}$ st

$$\Pr_{\text{IND-CPA}(\mathcal{A})}\left[b' = b\right] - \frac{1}{2} \leq \Pr_{\text{OW}(\mathcal{I})}\left[y' = y\right]$$

Proof

Game hopping technique

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$h \leftarrow H(r);$
$s \leftarrow h \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game G :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$h \xleftarrow{\$} \{0,1\}^k;$
$s \leftarrow h \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game G′ :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$s \xleftarrow{\$} \{0,1\}^k;$
$h \leftarrow s \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^\ell;$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$s \xleftarrow{\$} \{0,1\}^k;$
$c^\star \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(c^\star);$
$y' \leftarrow [z \in L_H^{\mathcal{A}} \,|\, f_{pk}(z) = x];$
return y'

1. Prove a probability claim for each hop
2. Obtain security bound by combining claims
3. Check execution time of constructed adversary

Conditional equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \leftarrow H(r)$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

By the Fundamental Lemma

$$\Pr_{\text{IND-CPA}}\left[b' = b\right] - \Pr_{\mathbf{G}}\left[b' = b\right] \leq \Pr_{\mathbf{G}}\left[r \in L_H^{\mathcal{A}}\right]$$

Equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$s \xleftarrow{\$} \{0,1\}^{k}$;
$h \leftarrow s \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

By optimistic sampling

$$\Pr_{\mathbf{G}}\left[r \in L_H^{\mathcal{A}}\right] = \Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right] \qquad \Pr_{\mathbf{G}}[b' = b] = \Pr_{\mathbf{G}'}[b' = b] = \frac{1}{2}$$

Equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$s \xleftarrow{\$} \{0,1\}^{k}$;
$h \leftarrow s \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

By optimistic sampling

$$\Pr_{\text{IND-CPA}}[b' = b] - \tfrac{1}{2} \leq \Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right]$$

Reduction

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}()$;
$y \xleftarrow{\$} \{0,1\}^\ell$;
$y' \leftarrow \mathcal{I}(f_{pk}(y))$;
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c^\star \leftarrow x \,\|\, s$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
$y' \leftarrow [z \in L_H^{\mathcal{A}} \mid f_{pk}(z) = x]$;
return y'

$$\Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right] \leq \Pr_{\mathrm{OW}(\mathcal{I})}[y' = y]$$

Reduction

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}()$;
$y \xleftarrow{\$} \{0,1\}^\ell$;
$y' \leftarrow \mathcal{I}(f_{pk}(y))$;
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c^\star \leftarrow x \,\|\, s$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
$y' \leftarrow [z \in L_H^{\mathcal{A}} \mid f_{pk}(z) = x]$;
return y'

$$\Pr_{\text{IND-CPA}(\mathcal{A})}[b' = b] - \tfrac{1}{2} \leq \Pr_{\text{OW}(\mathcal{I})}[y' = y]$$

Plug-and-pray inverter

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$

Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}()$;
$y \xleftarrow{\$} \{0,1\}^\ell$;
$y' \leftarrow \mathcal{I}(f_{pk}(y))$;
return $y = y'$

Adversary $\mathcal{I}'(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$s \xleftarrow{\$} \{0,1\}^k$;
$c^\star \leftarrow x \,\|\, s$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
$i \xleftarrow{\$} [1..q_H]$;
$y' \leftarrow L_H^{\mathcal{A}}[i]$;
return y'

$$\Pr_{\text{IND-CPA}(\mathcal{A})}\left[b' = b\right] - \frac{1}{2} \leq q_H \Pr_{\text{OW}(\mathcal{I}')}\left[y' = y\right]$$

Optimal Asymmetric Encryption Padding

Encryption $\mathcal{E}_{\mathrm{OAEP}(pk)}(m)$:
$r \xleftarrow{\$} \{0,1\}^{k_0}$;
$s \leftarrow G(r) \oplus (m \| 0^{k_1})$;
$t \leftarrow H(s) \oplus r$;
return $\mathsf{f}_{pk}(s \| t)$

Decryption $\mathcal{D}_{\mathrm{OAEP}(sk)}(c)$:
$(s,t) \leftarrow \mathsf{f}_{sk}^{-1}(c)$;
$r \leftarrow t \oplus H(s)$;
if $([s \oplus G(r)]_{k_1} = 0^{k_1})$
then $\{m \leftarrow [s \oplus G(r)]^{k}; \}$
else $\{m \leftarrow \perp; \}$
return m

$\oplus$ exclusive or $\|$ concatenation $[\cdot]$ projection 0 zero bitstring

OAEP: provable security

Game IND-CCA($\mathcal{A}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$

Game SPDOW($\mathcal{I}$)
$(sk, pk) \leftarrow \mathcal{K}()$;
$y \xleftarrow{\$} \{0,1\}^{k_2}$; $z \xleftarrow{\$} \{0,1\}^{k_3}$;
$x^\star \leftarrow \mathsf{f}_{pk}(y \,\|\, z)$;
$Y' \leftarrow \mathcal{I}(x^\star)$;
return $(y \in Y')$

FOR ALL IND-CCA adversary $\mathcal{A}$ against $(\mathcal{K}, \mathcal{E}_{\mathrm{OAEP}}, \mathcal{D}_{\mathrm{OAEP}})$,
THERE EXISTS a SPDOW adversary $\mathcal{I}$ against $(\mathcal{K}, \mathsf{f}, \mathsf{f}^{-1})$ st

$$\left|\Pr_{\text{IND-CCA}(\mathcal{A})}[b' = b] - \tfrac{1}{2}\right| \leq$$

$$\Pr_{\text{SPDOW}(\mathcal{I})}[y \in Y'] + \frac{3q_D q_G + q_D^2 + 4q_D + q_G}{2^{k_0}} + \frac{2q_D}{2^{k_1}}$$

and

$$t_{\mathcal{I}} \leq t_{\mathcal{A}} + q_D\, q_G\, q_H\, T_{\mathsf{f}}$$

Key length

- Estimation: factoring RSA-768 takes 2^{67} operations
- Extrapolation: factoring N-bit RSA modulus takes time $\propto$

$$\exp((1.9229 + o(1)) \log(N)^{1/3} \log(\log(N))^{2/3})$$

Modulus size	Number of Operations
512	2^{58}
768	2^{67}
1024	2^{77}
2048	2^{107}
3072	2^{129}
4096	2^{147}
5120	2^{162}
6144	2^{176}
7680	2^{193}
8192	2^{199}
15360	2^{259}

Practical interpretation for RSA-OAEP

- Reduction from PDOW to OW. Let $\ell < 2k$

$$\mathbf{Succ}_{RSA}^{\mathrm{SPDOW}_k^q}(t)\left(\mathbf{Succ}_{RSA}^{\mathrm{SPDOW}_k^q}(t) - 2^{\ell-2k+6}\right) \leq \mathbf{Succ}_{RSA}^{\mathrm{OW}_{2k}}(2t+q^2\ell^3)$$

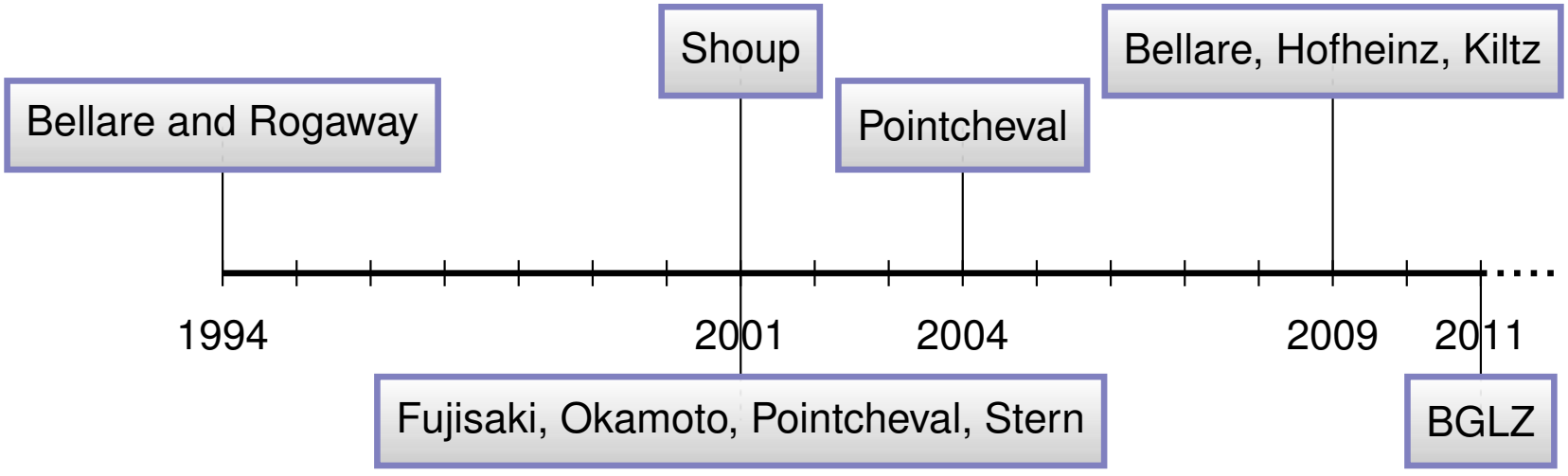
- Set bounds to adversary queries

$$q_D \leq 2^{30} \qquad\qquad q_G, q_H \leq 2^{60}$$

- Derive recommended (overly conservative) key size

4096

OAEP: provable security



1994 Purported proof of chosen-ciphertext security

2001 1994 proof gives weaker security; desired security holds

- for a modified scheme
- under stronger assumptions

2004 Filled gaps in 2001 proof

2009 Security definition needs to be clarified

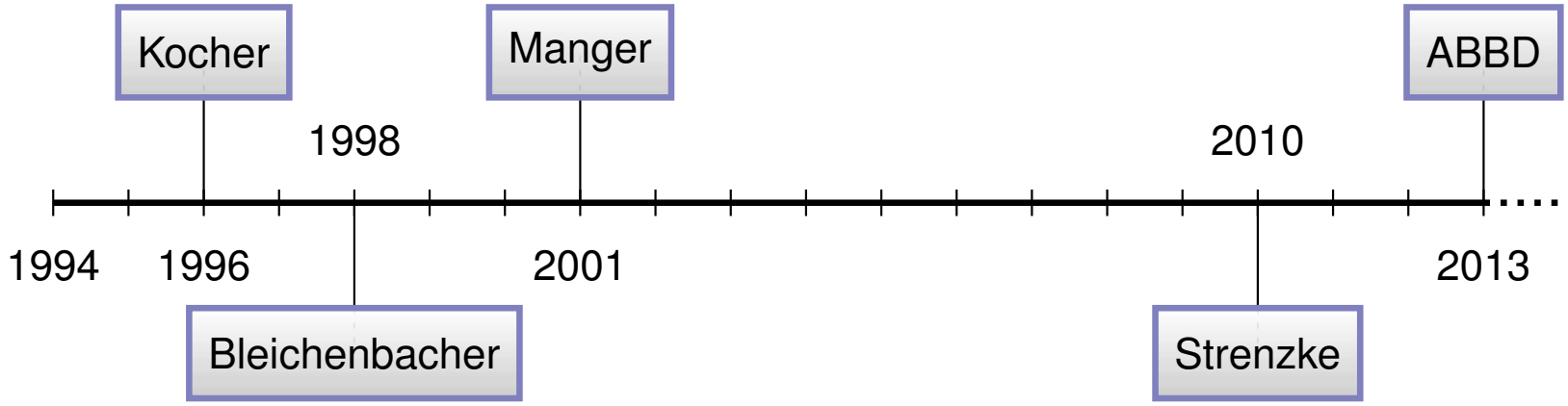
2011 Fills gaps in 2004 proof

Implementation of OAEP

Decryption $\mathcal{D}_{\mathrm{OAEP}(sk)}(c)$:
$(s,t) \leftarrow \mathsf{f}_{sk}^{-1}(c)$;
$r \leftarrow t \oplus H(s)$;
if $([s \oplus G(r)]_{k_1} = 0^{k_1})$
then $\{m \leftarrow [s \oplus G(r)]^k;\}$
else $\{m \leftarrow \perp;\}$
return m

Decryption $\mathcal{D}_{\mathrm{PKCS\text{-}C}(sk)}(res, c)$:
if $(c \in \mathrm{MsgSpace}(sk))$ then
$\{ (b0, s, t) \leftarrow \mathsf{f}_{sk}^{-1}(c)$;
$h \leftarrow MGF(s, hL)$; $i \leftarrow 0$;
while $(i < hLen + 1)$
$\{ s[i] \leftarrow t[i] \oplus h[i]$; $i \leftarrow i + 1$; $\}$
$g \leftarrow MGF(r, dbL)$; $i \leftarrow 0$;
while $(i < dbLen)$
$\{ p[i] \leftarrow s[i] \oplus g[i]$; $i \leftarrow i + 1$; $\}$
$l \leftarrow payload\_length(p)$;
if $(b0 = 0^8 \wedge [p]_l^{hLen} = 0..01 \wedge$
$[p]_{hLen} = LHash)$
then
$\{rc \leftarrow Success$;
$memcpy(res, 0, p, dbLen - l, l);\}$
else $\{rc \leftarrow DecryptionError;\}\}$
else $\{rc \leftarrow CiphertextTooLong;\}$
return rc;

OAEP: real-world security



Attacks from observing

- error messages
- execution time
  - ☞ RSA implementations
  - ☞ conversion from integers to bitstrings
    (RSA operates on strict subset of $\left[0..2^k\right]$)

Problems with cryptographic proofs

Unverifiable proofs

- Proofs are long and error-prone
- Rely on unstated and unverified invariants
- Intricate reasoning steps justified informally

[...] many proofs in cryptography have become essentially unverifiable. Our field may be approaching a crisis of rigor.
Bellare and Rogaway, 2004-2006

Abstraction gap

- Provable security reasons about algorithmic descriptions
- Standards constrain implementations
- Attackers target executable code

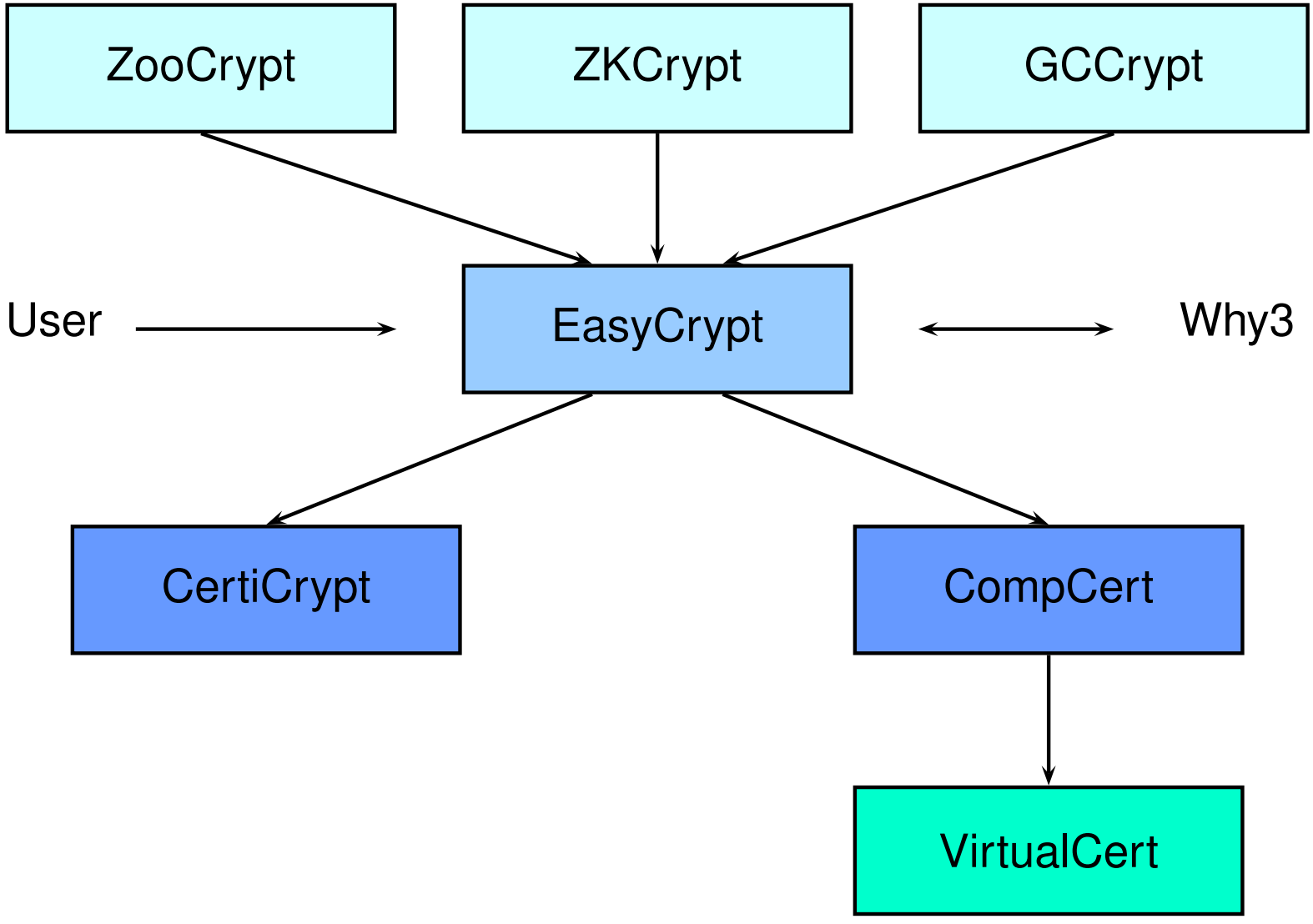
Real-world crypto is breakable; is in fact being broken; is one of many ongoing disaster areas in security. Bernstein, 2013

Computer-aided cryptographic proofs

Provable security as deductive relational verification of open probabilistic parametrized programs

- High-confidence reductionist proofs
  - ☞ machine-checked, independently verifiable proofs
  - ☞ adhere to cryptographic practice (same formalisms, guarantees and proof techniques)
- Manage complexity of real-world cryptography
- Increase confidence in implementations
  - ☞ minimize gap between proofs and code
  - ☞ prove effectiveness of countermeasures (on source code and machine code)
- Leverage existing verification techniques and tools
  - ☞ program logics, VC generation, invariant generation
  - ☞ SMT solvers, theorem provers, proof assistants
  - ☞ verified compilers

EasyCrypt toolchain



A language for cryptographic games

$\mathcal{C}$	::=	skip	skip
	|	$\mathcal{V} \leftarrow \mathcal{E}$	assignment
	|	$\mathcal{V} \xleftarrow{\$} \mathcal{D}$	random sampling
	|	$\mathcal{C};\ \mathcal{C}$	sequence
	|	if $\mathcal{E}$ then $\mathcal{C}$ else $\mathcal{C}$	conditional
	|	while $\mathcal{E}$ do $\mathcal{C}$	while loop
	|	$\mathcal{V} \leftarrow \mathcal{P}(\mathcal{E}, \ldots, \mathcal{E})$	procedure call

- $\mathcal{E}$: (higher-order) expressions
- $\mathcal{D}$: discrete sub-distributions
- $\mathcal{P}$: procedures

} user extensible

. oracles: concrete procedures

. adversaries: constrained abstract procedures

Program semantics

- A discrete sub-distribution over A is a map $\mu : A \to [0, 1]$ st.
  - ☞ $\sum_{a \in A} \mu(a) \leq 1$
  - ☞ $\mathrm{supp}(\mu) = \{a \in A \mid \mu(a) > 0\}$ is discrete
- Programs as sub-distribution transformers: $[\![c]\!]$
  - ☞ takes as input a memory m
  - ☞ returns a discrete sub-distribution over memories
- Probability of an event

$$\Pr_{c,m}[E] = \sum_{m' \in E} [\![c]\!]_m \; m'$$

Given $f : A \to [0, 1]$, define

$$\mu^\star(f) = \sum_{a \in A} f(a) \times \mu(\mathbb{1}_a)$$

Program semantics

$$[\![x \leftarrow e]\!]\, m = \mathbb{1}_{\{[m|x \mapsto [\![e]\!]\, m]\}}$$

$$[\![x \xleftarrow{\$} \mu]\!]\, m = \sum_{d \in \mathrm{dom}([\![\mu]\!]\, m)} \mathbb{1}_{\{[m|x \mapsto d]\}}$$

$$[\![c_1 ; c_2]\!]\, m = [\![c_2]\!]^\star\, ([\![c_1]\!]\, m)$$

$$[\![\text{if } b \text{ then } c_1 \text{ else } c_2]\!]\, m = \text{if } [\![b]\!]\, m \text{ then } [\![c_1]\!]\, m \text{ else } [\![c_2]\!]\, m$$

$$[\![\text{while } b \text{ do } c]\!] = \mathrm{fix}\, (\lambda f.\ \lambda m.\ \text{if } [\![b]\!]\, m \text{ then } f^\star\, ([\![c]\!]\, m) \text{ else } \mathbb{1}_{\{m\}})$$

Alternative characterization

$$[\![\text{while } e \text{ do } c]\!]\, m = \lambda f.\ \sup_{n \in \mathbb{N}} ([\![[\text{while } e \text{ do } c]_n]\!]\, m\ f)$$

where

$$\begin{aligned} [\text{while } e \text{ do } c]_0 &= \text{abort} \\ [\text{while } e \text{ do } c]_{n+1} &= \text{if } e \text{ then } (c;\ [\text{while } e \text{ do } c]_n) \end{aligned}$$

Reasoning about cryptographic games

- Probablistic Relational Hoare Logic
- Probabilistic Hoare Logic
- Program optimizations
- Ambient logic (inc. quantification over modules) allows
  - ☞ hybrid arguments
  - ☞ modular proofs
  - ☞ meta-arguments

(Deductive) program verification

- Art of proving that programs are correct
- Origins:
  - ☞ axiomatic semantics (Hoare'69)
  - ☞ weakest precondition calculus (Floyd'67)
- Major advances in:
  - ☞ language coverage
  - ☞ automation
  - ☞ proof engineering

Hoare logic

- Judgments $c : P \Longrightarrow Q$
  (P and Q are f.o. formulae over program variables)
- A judgment $c : P \Longrightarrow Q$ is valid iff

$$\forall m,\ m \models P \Rightarrow [\![c]\!]m = m' \Rightarrow m' \models Q$$

Selected rules

$$\frac{c : P \Longrightarrow Q \quad P' \Rightarrow P \quad Q \Rightarrow Q'}{c : P' \Longrightarrow Q'} \qquad \frac{}{x \leftarrow e : Q[e/x] \Longrightarrow Q}$$

$$\frac{c_1 : P \Longrightarrow Q \quad c_2 : Q \Longrightarrow R}{c_1 ; c_2 : P \Longrightarrow R}$$

$$\frac{c_1 : P \wedge e \Longrightarrow Q \quad c_2 : P \wedge \neg e \Longrightarrow Q}{\text{if } e \text{ then } c_1 \text{ else } c_2 : P \Longrightarrow Q}$$

$$\frac{c : I \wedge e \Longrightarrow I}{\text{while } e \text{ do } c : I \Longrightarrow I \wedge \neg e}$$

Verification condition generation

- Generate a set of verification conditions from annotated command and postcondition
- If all VCs are valid and $P \Rightarrow \mathrm{wp}(c, Q)$ then $c : P \Longrightarrow Q$

Selected rules

$$\begin{aligned}
\mathrm{wp}(x \leftarrow e, Q) &= Q[e/x] \\
\mathrm{wp}(c_1; c_2, R) &= \mathrm{wp}(c_1, \mathrm{wp}(c_2, R)) \\
\mathrm{wp}(\text{if } e \text{ then } c_1 \text{ else } c_2, Q) &= e \Rightarrow \mathrm{wp}(c_1, Q) \wedge \neg e \Rightarrow \mathrm{wp}(c_2, Q) \\
\mathrm{wp}(\text{while}_I\ e \text{ do } c, Q) &= I
\end{aligned}$$

The while rule generates two proof obligations

$$I \wedge e \Rightarrow \mathrm{wp}(c, I) \qquad I \wedge \neg e \Rightarrow Q$$

Beyond safety

2-safety and 2-programs safety

- 2 executions of the same program: information flow
- 2 programs: program equivalence

- Judgments

$$\vDash \{P\}\ c_1\ \sim\ c_2\ \{Q\}$$

(P and Q are f.o. formulae over tagged program variables)

- Validity

$$\forall m_1, m_2.\ (m_1, m_2) \vDash P \implies (\llbracket c_1 \rrbracket\ m_1, \llbracket c_2 \rrbracket\ m_2) \vDash Q$$

- Verification methods
  - ☞ embedding into Hoare logic
  - ☞ relational Hoare logic

pRHL: probabilistic relational Hoare logic

- Judgment

$$\models \{P\}\ c_1\ \sim\ c_2\ \{Q\}$$

  where P and Q denote relations on memories

- Validity

$$\forall m_1, m_2.\ (m_1, m_2) \models P \implies (\llbracket c_1 \rrbracket\ m_1, \llbracket c_2 \rrbracket\ m_2) \models Q^\sharp$$

- Definition of $\cdot^\sharp$ drawn from probabilistic process algebra

Deriving probability claims

Assume $\vDash \{P\}\ c_1 \sim c_2\ \{Q\}$ and $(m_1, m_2) \models P$

Equivalence

- If $Q \stackrel{\text{def}}{=} \bigwedge_{x \in X} x\langle 1 \rangle = x\langle 2 \rangle$ and $\mathrm{FV}(A) \subseteq X$ then

$$\Pr_{c_1, m_1}[A] = \Pr_{c_2, m_2}[A]$$

- If $Q \stackrel{\text{def}}{=} A\langle 1 \rangle \Leftrightarrow B\langle 2 \rangle$ then

$$\Pr_{c_1, m_1}[A] = \Pr_{c_2, m_2}[B]$$

Conditional equivalence

- If $Q \stackrel{\text{def}}{=} \neg F\langle 2 \rangle \Rightarrow \bigwedge_{x \in X} x\langle 1 \rangle = x\langle 2 \rangle$ and $\mathrm{FV}(A) \subseteq X$ then

$$\Pr_{c_1, m_1}[A] - \Pr_{c_2, m_2}[A] \leq \Pr_{c_2, m_2}[F]$$

- If $Q \stackrel{\text{def}}{=} \neg F\langle 2 \rangle \Rightarrow (A\langle 1 \rangle \Leftrightarrow B\langle 2 \rangle)$ then

$$\Pr_{c_1, m_1}[A] - \Pr_{c_2, m_2}[B] \leq \Pr_{c_2, m_2}[F]$$

Lifting Relations to sub-distributions

Existential definition

$(\mu_1, \mu_2) \models Q^\sharp$ iff there exists $\mu \in \mathcal{D}(\mathcal{M} \times \mathcal{M})$ s.t.

- $\pi_i(\mu) = \mu_i$, where $\pi_1(\mu)\ (a) = \sum_{b \in B} \mu(a, b)$
- $\mathrm{supp}(\mu) \subseteq Q$, i.e. $\mu(a, b) > 0 \Rightarrow Q(a, b)$

Inductive definition

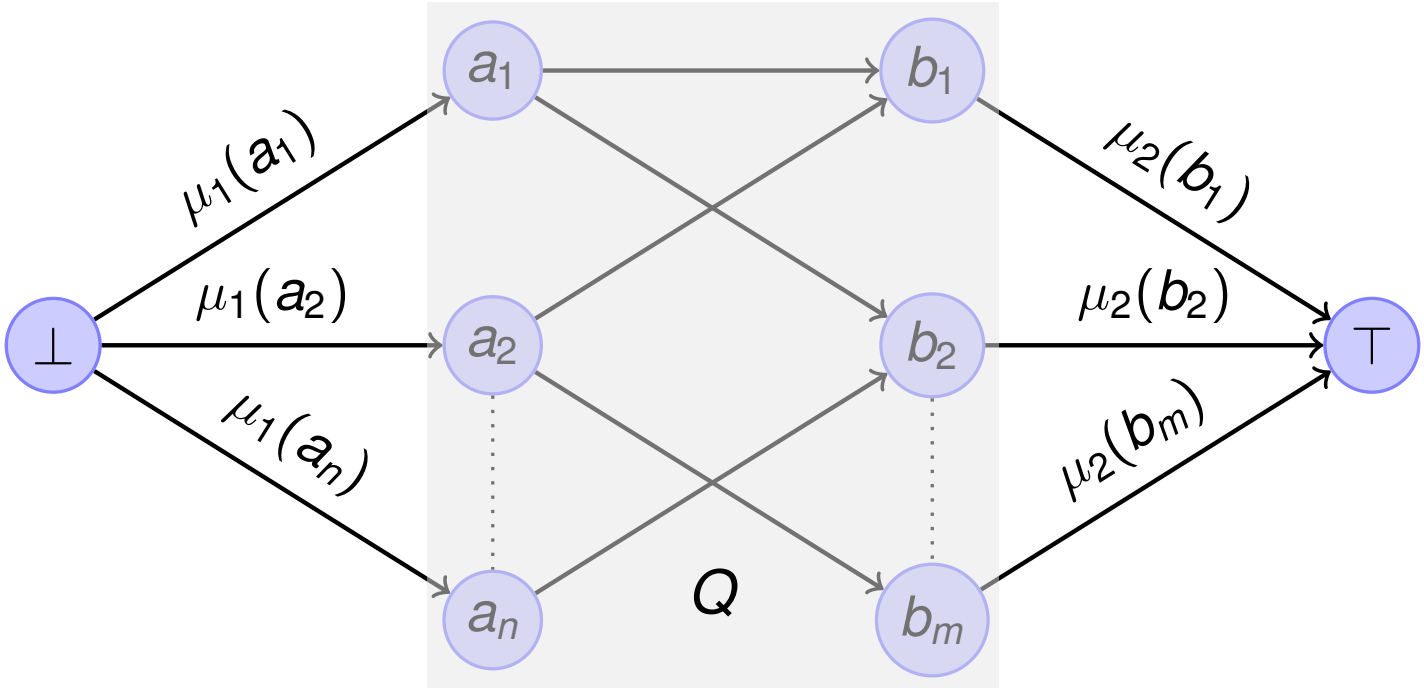
- If $(s, t) \models Q$ then $(\delta_s, \delta_t) \models Q^\sharp$
- If $(\mu_i, \nu_i) \models Q^\sharp$ and $\sum_i p_i = 1$, then

$$\left(\sum_i p_i \, \mu_i, \sum_i p_i \, \nu_i \right) \models Q^\sharp$$

Flow network definition

$(\mu_1, \mu_2) \models Q^\sharp$ iff the maximum flow in the induced network is 1

Flow networks



Proof rules: random assignment

Intuition

Let A be a finite set and let $f, g : A \to B$. Define

- $c = x \xleftarrow{\$} \mu; y \leftarrow f\ x$
- $c' = x \xleftarrow{\$} \mu'; y \leftarrow g\ x$

Then $[\![c]\!] = [\![c']\!]$ (extensionally) iff there exists $h : A \overset{1-1}{\to} A$ st

- $f = g \circ h$
- for all a, $\mu(a) = \mu'(h(a))$

$$\frac{h \text{ is 1-1 and } \forall a,\ \mu(a) = \mu'(h(a))}{\vDash \{\forall v, Q[h\ v/x\langle 1\rangle][v/x\langle 2\rangle]\}\ \ x \xleftarrow{\$} \mu\ \sim\ x \xleftarrow{\$} \mu'\ \ \{Q\}}$$

- Rule captures a special case of lifting
- General rule might lead to untractable arithmetic equalities

Proof rule: assignments and conditionals

Assignments

$$\frac{}{\vDash \{Q[e\langle 1\rangle / x\langle 1\rangle][e'\langle 2\rangle / x'\langle 2\rangle]\}\ x \leftarrow e\ \sim\ x' \leftarrow e'\ \{Q\}}$$

$$\frac{}{\vDash \{Q[x\langle 1\rangle := e\langle 1\rangle]\}\ x \leftarrow e\ \sim\ \text{skip}\ \{Q\}}$$

Conditionals

$$\frac{P\ \Rightarrow\ e\langle 1\rangle\ =\ e'\langle 2\rangle \qquad \vDash \{P \wedge e\langle 1\rangle\}\ c_1\ \sim\ c_1'\ \{Q\} \qquad \vDash \{P \wedge \neg e\langle 1\rangle\}\ c_2\ \sim\ c_2'\ \{Q\}}{\vDash \{P\}\ \text{if } e \text{ then } c_1 \text{ else } c_2\ \sim\ \text{if } e' \text{ then } c_1' \text{ else } c_2'\ \{Q\}}$$

$$\frac{\vDash \{P \wedge e\langle 1\rangle\}\ c_1\ \sim\ c\ \{Q\} \qquad \vDash \{P \wedge \neg e\langle 1\rangle\}\ c_2\ \sim\ c\ \{Q\}}{\vDash \{P\}\ \text{if } e \text{ then } c_1 \text{ else } c_2\ \sim\ c\ \{Q\}}$$

Loops

Two-sided rule

$$\frac{\begin{array}{c} P \;\Rightarrow\; e\langle 1\rangle \;=\; e'\langle 2\rangle \\ \vDash \{P \wedge e\langle 1\rangle\}\; c \;\sim\; c'\; \{P\} \end{array}}{\vDash \{P\}\ \text{while } e \text{ do } c \;\sim\; \text{while } e' \text{ do } c'\ \{P \wedge \neg e\langle 1\rangle\}}$$

- rule is incomplete: same number of iterations

One sided-rules

- standard rule with losslessness verification condition

Optimizations

- unrolling
- loop fission/fusion/. . .

Adversaries

$$\frac{\forall \mathcal{O}.\ \vDash \{Q \wedge =_W\}\ z \leftarrow \mathcal{O}(\vec{w})\ \sim\ z \leftarrow \mathcal{O}(\vec{w})\ \left\{Q \wedge =_{\{z\}}\right\}}{\vDash \{Q \wedge =_Y\}\ x \leftarrow \mathcal{A}(\vec{y})\ \sim\ x \leftarrow \mathcal{A}(\vec{y})\ \left\{Q \wedge =_{\{x\}}\right\}}$$

- Adversaries perform arbitrary sequences of oracle calls (and intermediate computations)
- No functional specification
- Given the same inputs, provide the same outputs

Reasoning about Failure Events

Lemma (Fundamental Lemma)

Let A, B, F be events and G_1, G_2 be two games such that

$$\Pr_{G_1}[A \wedge \neg F] = \Pr_{G_2}[B \wedge \neg F]$$

Then

$$\left|\Pr_{G_1}[A] - \Pr_{G_2}[B]\right| \leq \max(\Pr_{G_1}[F], \Pr_{G_2}[F])$$

Fundamental Lemma: syntactic criterion

Let $A = B$ and $F =$ bad. If G_0 and G_1 are syntactically identical until bad is set, e.g.

Game G_0 :	**Game** G_1 :
...	...
bad $\leftarrow$ true; c_0	bad $\leftarrow$ true; c_1
...	...

then

- $\Pr_{G_1,m}[A \mid \neg\text{bad}] = \Pr_{G_2,m}[A \mid \neg\text{bad}]$
- $\Pr_{G_1,m}[\text{bad}] = \Pr_{G_2,m}[\text{bad}]$

Corollary

$$|\Pr_{G_1,m}[A] - \Pr_{G_2,m}[A]| \leq \Pr_{G_{1,2}}[\text{bad}]$$

Fundamental Lemma: syntactic criterion

Let $A = B$ and $F = \text{bad}$. If G_0 and G_1 are syntactically identical until bad is set, e.g.

Game G_0 :	**Game** G_1 :
...	...
bad $\leftarrow$ true; c_0	bad $\leftarrow$ true; c_1
...	...

then

- $\Pr_{G_1,m}[A \mid \neg\text{bad}] = \Pr_{G_2,m}[A \mid \neg\text{bad}]$
- $\Pr_{G_1,m}[\text{bad}] = \Pr_{G_2,m}[\text{bad}]$

Corollary

$$|\Pr_{G_1,m}[A] - \Pr_{G_2,m}[A]| \leq \Pr_{G_{1,2}}[\text{bad}]$$

Fundamental Lemma in pRHL

Let A, B, F be events and G_1, G_2 be two games such that

$$\vDash \{\text{true}\}\ G_1 \sim G_2\ \{(F\langle 1\rangle \leftrightarrow F\langle 2\rangle) \wedge (\neg F\langle 2\rangle \rightarrow (A\langle 1\rangle \leftrightarrow B\langle 2\rangle))\}$$

then

$$\left|\Pr_{G_1}[A] - \Pr_{G_2}[B]\right| \leq \Pr_{G_2}[F] \qquad \Pr_{G_1}[F] = \Pr_{G_2}[F]$$

Simpler variants

$$\vDash \{\text{true}\}\ G_1 \sim G_2\ \{\neg F\langle 2\rangle \rightarrow (A\langle 1\rangle \leftrightarrow B\langle 2\rangle)\}$$

then

$$\left|\Pr_{G_1}[A] - \Pr_{G_2}[B]\right| \leq \Pr_{G_2}[F]$$

Even simpler

$$\vDash \{\text{true}\}\ G_1 \sim G_2\ \{\neg F\langle 2\rangle \rightarrow B\langle 2\rangle \rightarrow A\langle 1\rangle\}$$

then

$$\Pr_{G_1}[A] - \Pr_{G_2}[B] \leq \Pr_{G_2}[F]$$

Fundamental lemma: adversary

For an adversary $\mathcal{A}$ with oracle access to oracles $\mathcal{O}^i$, it suffices to check that:

- F is monotonic

$$\{F\}\ \mathcal{O}_1^i\ \{F\} \qquad\qquad \{F\}\ \mathcal{O}_2^i\ \{F\}$$

- Oracle calls preserve equivalence up to failure

$$\models y \leftarrow \mathcal{A}^{\mathcal{O}_1^i}(\vec{x}) \sim y \leftarrow \mathcal{A}^{\mathcal{O}_2^i}(\vec{x}) :$$
$$(\neg F\langle 2\rangle \wedge Q \wedge \vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle) \Longrightarrow (\neg F\langle 2\rangle \rightarrow Q \wedge y\langle 1\rangle = y\langle 2\rangle)$$

pHL: probabilistic Hoare logic

$$[c : P \Longrightarrow Q] \leq \delta \qquad [c : P \Longrightarrow Q] = \delta \qquad [c : P \Longrightarrow Q] \geq \delta$$

- P, Q predicates on memories
- δ a real expression evaluated on initial memory

Interpretation $(\leq)$

$$\forall m,\ m \models P \Rightarrow [\![c]\!] m\, Q \leq \delta\, m$$

Sample rules

Assignment

$$[x \leftarrow e : P[e/x] \Longrightarrow P] = 1$$

Sequential composition for $\geq$

$$\frac{[c_1 : P \Longrightarrow R] \geq \delta_1 \qquad [c_2 : R \Longrightarrow Q] \geq \delta_2}{[c_1 ; c_2 : P \Longrightarrow Q] \geq \delta_1 \delta_2}$$

Sequential composition for $\leq$

$$\frac{\begin{array}{cc} [c_1 : P \Longrightarrow R] \leq \delta_1 & [c_2 : R \Longrightarrow Q] \leq \delta_2 \\ [c_1 : P \Longrightarrow \neg R] \leq \delta_3 & [c_2 : \neg R \Longrightarrow Q] \leq \delta_4 \end{array}}{[c_1 ; c_2 : P \Longrightarrow Q] \leq \delta_1 \delta_2 + \delta_3 \delta_4}$$

Applications of probabilistic Hoare Logic

Let P be a precondition.

- Termination wrt P:

$$[c : P \Longrightarrow \top] = 1$$

- Cost:

$$[\bar{c} : P \Longrightarrow cost \leq p] = 1$$

  where $\bar{c}$ is an annotated version of c

- Observational equivalence $c_1 \equiv_{P,=_{\vec{x}}} c_2$

$$\forall \vec{a}\, p,\ \left[c_1 : P \Longrightarrow \vec{x} = \vec{a}\right] \leq p \Leftrightarrow \left[c_2 : P \Longrightarrow \vec{x} = \vec{a}\right] \leq p$$

Example: Bellare and Rogaway 1993 encryption

Game IND-CPA($\mathcal{A}$) :
$(sk, pk) \leftarrow \mathcal{K}(\ )$;
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk)$;
$b \xleftarrow{\$} \{0,1\}$;
$c^\star \leftarrow \mathcal{E}_{pk}(m_b)$;
$b' \leftarrow \mathcal{A}_2(c^\star)$;
return $(b' = b)$

Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell$;
$s \leftarrow H(r) \oplus m$;
$y \leftarrow f_{pk}(r) \,\|\, s$;
return y

For every IND-CPA adversary $\mathcal{A}$, there exists an inverter $\mathcal{I}$ st

$$\Pr_{\text{IND-CPA}(\mathcal{A})}\left[b' = b\right] - \frac{1}{2} \leq \Pr_{\text{OW}(\mathcal{I})}\left[y' = y\right]$$

Proof

Game hopping technique

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$h \leftarrow H(r);$
$s \leftarrow h \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game G :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$h \xleftarrow{\$} \{0,1\}^k;$
$s \leftarrow h \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game G′ :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$s \xleftarrow{\$} \{0,1\}^k;$
$h \leftarrow s \oplus m;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^\ell;$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$s \xleftarrow{\$} \{0,1\}^k;$
$c^\star \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(c^\star);$
$y' \leftarrow [z \in L_H^{\mathcal{A}} \,|\, f_{pk}(z) = x];$
return y'

1. For each hop
   - prove validity of pRHL judgment
   - derive probability claims
   - (possibly) resolve some probability expressions using pHL
2. Obtain security bound by combining claims
3. Check execution time of constructed adversary

Conditional equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \leftarrow H(r)$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

$$\models \{\top\} \text{ IND-CPA } \sim \mathbf{G} \left\{ (\neg r \in L_H^{\mathcal{A}})\langle 2 \rangle \rightarrow =_{b,b'} \right\}$$

$$\Pr_{\text{IND-CPA}}\left[b' = b\right] - \Pr_{\mathbf{G}}\left[b' = b\right] \leq \Pr_{\mathbf{G}}\left[r \in L_H^{\mathcal{A}}\right]$$

Equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$s \xleftarrow{\$} \{0,1\}^{k}$;
$h \leftarrow s \oplus m$;
$c \leftarrow f_{pk}(r) \| s$;
return c

$$\models \{\top\}\ \mathbf{G}\ \sim\ \mathbf{G}'\ \left\{=_{b,b',r,L_H^{\mathcal{A}}}\right\}$$

$$\Pr_{\mathbf{G}}\left[r \in L_H^{\mathcal{A}}\right] = \Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right] \qquad \Pr_{\mathbf{G}}[b' = b] = \Pr_{\mathbf{G}'}[b' = b] = \frac{1}{2}$$

Equivalence

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$h \xleftarrow{\$} \{0,1\}^{k}$;
$s \leftarrow h \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

$\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^{\ell}$;
$s \xleftarrow{\$} \{0,1\}^{k}$;
$h \leftarrow s \oplus m$;
$c \leftarrow f_{pk}(r) \,\|\, s$;
return c

$$\models \{\top\}\ \mathbf{G}\ \sim\ \mathbf{G}'\ \left\{=_{b,b',r,L_H^{\mathcal{A}}}\right\}$$

$$\Pr_{\text{IND-CPA}}[b' = b] - \tfrac{1}{2} \leq \Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right]$$

Reduction

Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$s \xleftarrow{\$} \{0,1\}^k;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^\ell;$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$s \xleftarrow{\$} \{0,1\}^k;$
$c^\star \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(c^\star);$
$y' \leftarrow [z \in L_H^{\mathcal{A}} \mid f_{pk}(z) = x];$
return y'

$$\models \{\top\}\ \mathbf{G}'\ \sim\ \mathrm{OW}\ \left\{(r \in L_H^{\mathcal{A}})\langle 1\rangle \rightarrow (y' = y)\langle 2\rangle\right\}$$

$$\Pr_{\mathbf{G}'}\left[r \in L_H^{\mathcal{A}}\right] \leq \Pr_{\mathrm{OW}(\mathcal{I})}[y' = y]$$

Reduction

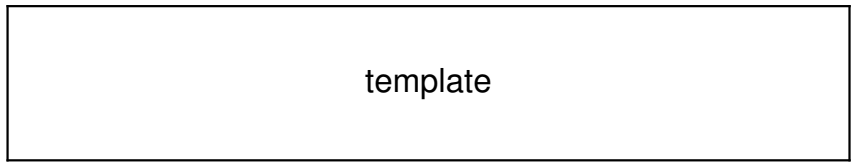
Game INDCPA :
$(sk, pk) \leftarrow \mathcal{K}();$
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$c^\star \leftarrow \mathcal{E}_{pk}(m_b);$
$b' \leftarrow \mathcal{A}_2(c^\star);$
return $(b' = b)$
Encryption $\mathcal{E}_{pk}(m)$:
$r \xleftarrow{\$} \{0,1\}^\ell;$
$s \xleftarrow{\$} \{0,1\}^k;$
$c \leftarrow f_{pk}(r) \,\|\, s;$
return c

Game OW :
$(sk, pk) \leftarrow \mathcal{K}();$
$y \xleftarrow{\$} \{0,1\}^\ell;$
$y' \leftarrow \mathcal{I}(f_{pk}(y));$
return $y = y'$
Adversary $\mathcal{I}(x)$:
$(m_0, m_1) \leftarrow \mathcal{A}_1(pk);$
$b \xleftarrow{\$} \{0,1\};$
$s \xleftarrow{\$} \{0,1\}^k;$
$c^\star \leftarrow x \,\|\, s;$
$b' \leftarrow \mathcal{A}_2(c^\star);$
$y' \leftarrow [z \in L_H^{\mathcal{A}} \mid f_{pk}(z) = x];$
return y'

$$\models \{\top\}\ \mathbf{G}'\ \sim\ \mathrm{OW}\ \left\{(r \in L_H^{\mathcal{A}})\langle 1\rangle \rightarrow (y' = y)\langle 2\rangle\right\}$$

$$\Pr_{\text{IND-CPA}(\mathcal{A})}[b' = b] - \tfrac{1}{2} \leq \Pr_{\text{OW}(\mathcal{I})}[y' = y]$$

Variants of OAEP



Automated proofs and exploration

The next 700 cryptosystems (after Landin, 1966)

Do the cryptosystems reflect [...] the situations that are being catered for? Or are they accidents of history and personal background that may be obscuring fruitful developments?
[...]
We must think in terms, not of cryptosystems, but of families of cryptosystems. That is to say we must systematize their design so that a new system is a point chosen from a well-mapped space, rather than a laboriously devised construction.

The two views of cryptography

Computational cryptography

- Strong ties with complexity theory
- Feasible adversary breaks scheme with small probability
- Design of secure and efficient primitives and protocols
- Complex and manual proofs

The two views of cryptography

Computational cryptography

- Strong ties with complexity theory
- Feasible adversary breaks scheme with small probability
- Design of secure and efficient primitives and protocols
- Complex and manual proofs

Symbolic cryptography (Dolev and Yao, 1983)

- Assume perfect cryptography
- Adversary cannot win
- Discovery of logical bugs in protocols
- Strong ties with verification
- Automated proofs

Reconciling the two views

(Abadi and Rogaway, 2000)

Computational soundness

Security in symbolic model implies computational security

- . . . under non-standard assumptions on primitives
- Symbolic tools deliver asymptotic guarantees
- . . . but no concrete guarantees
- Applicable to many settings
- . . . but some impossibility results

Our approach

Judgment

$$c :_p \varphi$$

- Reasons about probability of events
- Concrete probability can be computed
  - ☞ on the fly
  - ☞ a posteriori (judgments use only $p = 0, \frac{1}{2}$)
- Side-conditions are discharged by symbolic methods
  - ☞ what is the entropy of e?
  - ☞ can I compute e' from e?

Also use symbolic methods for

- Finding attacks
- Computing decryption algorithm

An algebraic view of padding-based schemes

Encryption algorithms are modelled as algebraic expressions

$\mathcal{E}$	$::=$	m	input message
	$\mid$	0	zero bitstring
	$\mid$	$\mathcal{R}$	uniform random bitstring
	$\mid$	$\mathcal{E} \oplus \mathcal{E}$	xor
	$\mid$	$\mathcal{E} \parallel \mathcal{E}$	concatenation
	$\mid$	$[\mathcal{E}]_s^s$	projection
	$\mid$	$H(\mathcal{E})$	hash
	$\mid$	$f(\mathcal{E})$	trapdoor permutation

Decryption algorithms are modelled using list comprehension

$$\vec{x} \xleftarrow{c} \vec{L}_H^{\mathcal{A}} : T \rhd e$$

where $\quad T ::= e = e \mid e \in L_H \mid e \in L_H^{\mathcal{A}} \mid T \wedge T$

Semantics

Left-to-right evaluation with sharing, yields a pWHILE procedure

Example

$$f((G(r) \oplus (m \| 0)) \| H(G(r) \oplus (m \| 0)) \oplus r)$$

interpreted as:

$$\begin{array}{l} r \xleftarrow{\$} \{0,1\}^k; \\ g \leftarrow G(r); \\ s \leftarrow g \oplus (m \| 0); \\ h \leftarrow H(s); \\ \text{return } f_{pk}(s \| (h \oplus r)) \end{array}$$

Proof principles: chosen-plaintext security

Failure event	Replace $H(e)$ by fresh r
Optimistic sampling	Replace $e \oplus r$, where r is fresh, by r
Probability	Compute probability of $b = b'$ or $e \in L$
Reduction	Find inverter and apply one-wayness

plus a few additional rules

Deducibility

$$\frac{}{e \vdash e} \qquad \frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash e_1 \,\|\, e_2}[\text{Conc}] \qquad \frac{e \vdash e_1 \quad e \vdash e_2}{e \vdash e_1 \oplus e_2}[\text{Xor}]$$

$$\frac{e \vdash e'}{e \vdash [e']_n^{\ell}}[\text{Proj}] \qquad \frac{e \vdash e_1 \quad \vdash e_1 \doteq e_2}{e \vdash e_2}[\text{Conv}]$$

$$\frac{e \vdash e'}{e \vdash H(e')}[\text{H}] \qquad \frac{e \vdash e'}{e \vdash f(e')}[\text{F}] \qquad \boxed{\frac{e \vdash e'}{e \vdash f^{-1}(e')}[\text{Finv}]}$$

Convertibility

- Based on equational theory of bitstrings
- Decidable for probabilistic expressions without H, f, f^{-1}

Useful for

- Finding decryption algorithm and attacks
- Proof rules: symbolic entropy and symbolic reduction

Attack finding

- Apply correcteness check
  - ☞ is decryption possible with a key? $r \parallel f(r)$
- Apply simple filters, eg
  - ☞ is decryption possible without a key? $m \parallel f(r)$
  - ☞ is encryption randomized? $f(m)$
  - ☞ is randomness extractable without a key? $r \parallel f(m \oplus r)$
- Apply static equivalence

Proof system for IND-CPA

$$\frac{m \notin c^\star}{c^\star : \mathsf{Guess}}\,[\mathsf{Indep}] \qquad \frac{c^\star : \varphi \quad r \text{ fresh}}{(c^\star : \varphi)\{e \oplus r/r\}}\,[\mathsf{Opt}]$$

$$\frac{e \vdash \vec{r} \qquad \vec{r} \cap \mathcal{R}(c^\star) = \emptyset}{c^\star : e \in L_H^{\mathcal{A}}}\,[\mathsf{Indom}]$$

$$\frac{e \vdash^{\mathcal{A}} \vec{r} \qquad f(\vec{r}) \,\|\, \vec{r_0} \,\|\, m \vdash_{t'}^{\mathcal{A}} c^\star \qquad \vec{r} \cap \vec{r_0} = \emptyset}{c^\star : e \in L_H^{\mathcal{A}}}\,[\mathsf{OW}]$$

$$\frac{c^\star : \phi \quad c^\star : e \in_H^{\mathcal{A}} \quad r \notin \mathcal{R}(e) \quad H \notin \mathcal{H}(c^\star, \phi, e)}{c^\star[H(e)/r] : \varphi[H(e)/r]}\,[\mathsf{Fail}]$$

+ a few more rules that are seldom needed
(eg a variant of [Indom] to prove INDCPA of OAEP under OW)

Chosen-ciphertext security

Principles

Extensionality Replace e or d by equivalent ones

Plaintext awareness Reject invalid ciphertexts

Plaintext extractor "Public" decryption oracle can be eliminated

Currently restricted to non-programmable random oracles.

Attack finding

- is encryption malleable? $f(r) \| m \oplus G(r)$

Soundness

- Once and for all
  - \+ “global” guarantee
  - \+ avoids resorting to intermediate framework
- By generating a pRHL/EasyCrypt proof for each scheme
  - \+ limits Trusted Computing Base
  - \+ proofs can be combined and reused
  - \- currently restricted to IND-CPA

Systematic exploration

- Generate well-typed terms up to user-defined constraints
- Check for decryption algorithm and attacks
- Launch proof search (strategy + backtracking)
- Compile successful runs to EasyCrypt (for IND-CPA)
- Practical interpretation

Evaluation

- Precise (no spurious attack, few unknowns)
- Efficient (attacks and proofs found instantaneously)

Minimality in cryptography

- OAEP (1994):

$$f((m\|0) \oplus G(r) \;\|\; r \oplus H((m\|0) \oplus G(r)))$$

not that Optimal; needs redundancy

- SAEP (2001):

$$f(r \;\|\; (m\|0) \oplus G(r))$$

tighter reduction; needs redundancy

- ZAEP (with David Pointcheval):

$$f(r \;\|\; m \oplus G(r))$$

tighter reduction, bit-optimal, redundancy-free

ZAEP

For every INDCCA adversary $\mathcal{A}$ there exists an inverter $\mathcal{I}$ st

$$\left| \Pr_{\text{IND-CCA}}[b = b'] - \frac{1}{2} \right| \leq \mathbf{Succ}_f^{\text{OW}}(\mathcal{I}) + \frac{q_D}{2^n}$$

Based on existence of two efficient algorithms:

- CIE: given $f(r, s_1)$, $f(r, s_2)$ with $s_1 \neq s_2$, returns s_1, s_2 and r
- SIE: given $f(r, s)$ and r returns s

Algorithms exist for RSA with exponents 2 and 3

Provable security of executable code

Proof by reduction:

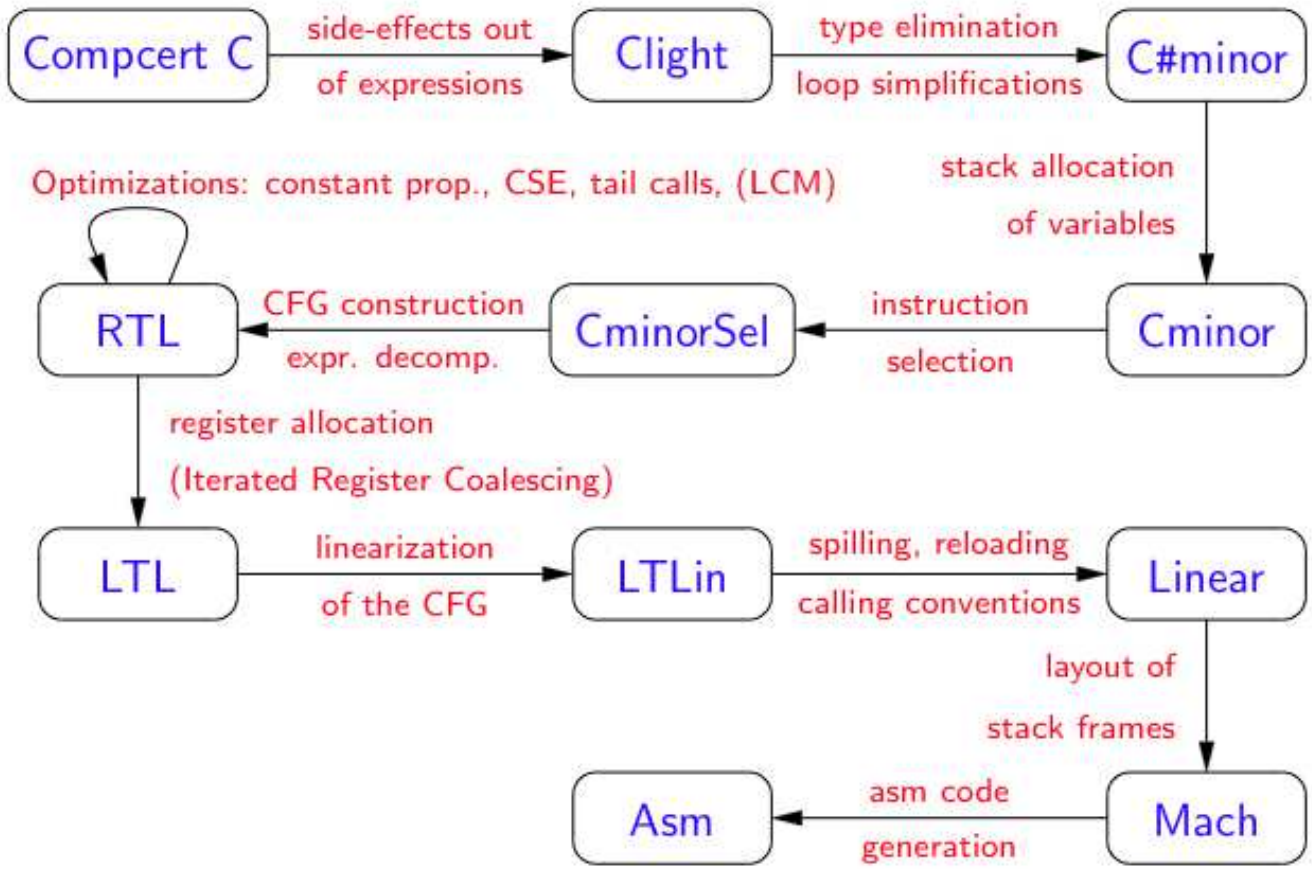
FOR ALL adversary that breaks the assembly code,
THERE EXISTS an adversary that breaks the source code

Proof relies on

- adversary model: low-level adversary does not observe computations (more than allowed in the source program)
- semantic preservation

CompCert (Leroy, 2006)

- Optimizing C compiler implemented in Coq
- Formal proof of semantic preservation



Semantic preservation in CompCert

Preservation of event traces

- system calls (“external calls”),
- I/O from and to the environment, and
- user-defined events

Issues with semantics preservation

Ideal (probabilistic) operations

- random sampling of bitstrings
- hash function (random oracle)

Libraries

- Implementations use arithmetic libraries

Approach

Proved semantics preservation using

- environment to model ideal operations
- trusted libraries to implement arithmetic

Side-channels and countermeasures

Well-known recipes for security disasters

- branching on secrets
- array accesses with secret indices

Want obliviousness

- control flow does not depend on secrets
- memory accesses do not depend on secrets

Problems

- algorithms might fail to satisfy obliviousness
- compilers might break both properties

Approach

- develop property-specific solutions
- formally verified using CompCert

Control flow obliviousness

- Prove security wrt “program counter model” game
  - ☞ add left/right tags to all branching statements
  - ☞ return traces to adversary
- Check compiler does not introduce new branches

Applied to implementation of OAEP based on LIP library

Memory obliviousness

- AES, DES. . . do not satisfy memory obliviousness
- Does compiler preserve memory obliviousness?

Solution

- Flow-sensitive information flow analysis of assembly code
- Stealth memory for accesses breaking obliviousness

Applied to implementations of AES

Case studies

- Encryption: OAEP, ZAEP, Hashed ElGamal, Cramer-Shoup, Boneh-Franklin IBE
- Signature: Full Domain Hash, BLS
- Hash functions:
  - ☞ Merkle-Damgard, Keccak
  - ☞ hashing into elliptic curves
- Zero knowledge protocols
- Authenticated Key Exchange
- Differential privacy

Conclusion

- Solid foundation for cryptographic proofs
- Formal verification of emblematic case studies
- Automated exploration of classes of constructions
- Discovery of new practical schemes
- Narrowing the gap between proofs and code

Cryptography is

- a thriving research area at the crossroads of many fields
- a great source of challenging problems
- an exciting opportunity to apply PL and PV techniques

`http://www.easycrypt.info`