Special Topics in Cryptography

Mohammad Mahmoody

Last time

- Secrecy based on (unproven) computational assumptions
- Pseudorandom generators
- How to encrypt longer messages in an ind-secure way using a PRG

Today

- How to make PRGs stretch more
- How to use Cryptographic Hash Functions to get PRGs
- Chosen plain-text security
- Pseudorandom generators (functions) -> CPA secure encryption

Recall: using PRGs to encrypt longer messages

- Key k of length n
- Message m of length $2n$
- A PRG $g: \{0,1\}^n \to \{0,1\}^{2n}$

Computationally bounded adversary

$\approx U_{2n}$

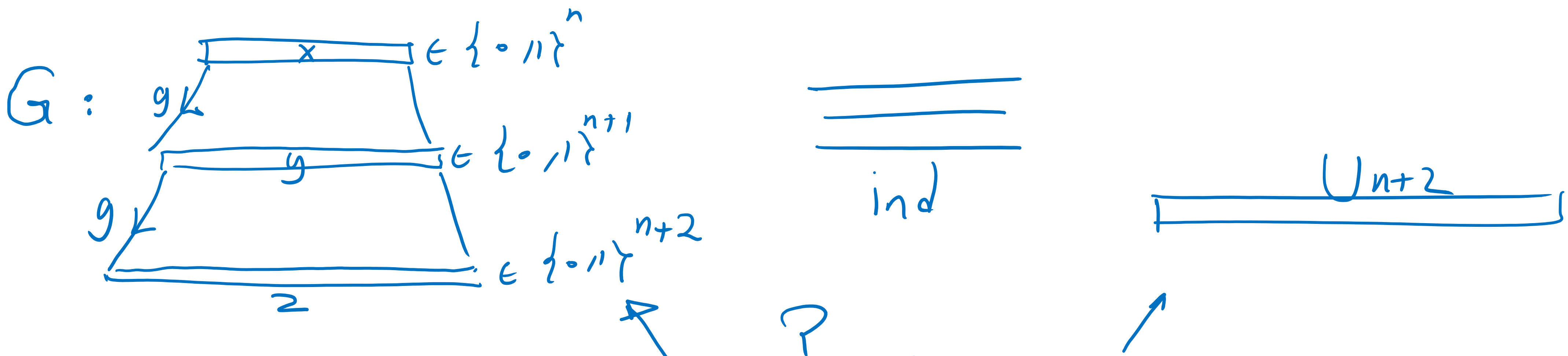
- $\mathrm{Enc}(k, m) = g(k) \oplus m$
- $\mathrm{Dec}(k, c) = c \oplus g(k)$

How to make PRGs stretch the output more?

Suppose we have $g:\{0,1\}^n \longrightarrow \{0,1\}^{n+1}$ and $g(U_n)$ is pseudo-random.

→ works for all $\underline{\underline{n}}$

Goal: how to get $G:\{0,1\}^n \longrightarrow \{0,1\}^{n+2}$: we want this for all n.



$$G(U_n) = g(g(U_n)) \overset{?}{\equiv} U_{n+2}$$

g: secure: ∀ poly time Adv. A $\quad g(U_n) \equiv_{\varepsilon(n)} U_{n+1}$

negligible

$|\Pr[A(g(U_n))=1] - \Pr[A(U_{n+1})=1]| \leq \varepsilon(n).$

Continuing Proof of security

Goal: Proving for all poly-time Adv B

$G(U_n) \equiv_{\delta(n)} U_{n+2}$ for some negligible $\delta(.)$.

Real World

$$\begin{tikzcd} \underbrace{G(U_n)}_{g_1 g_2(U_n)} \arrow[r, "y"] & B \arrow[d] \\ & b \end{tikzcd}$$

$P_R = \Pr_{\text{Real}}[A(y)=1]$

Hybrid World

$$\begin{tikzcd} g_1(U_{n+1}) \arrow[r] & y \arrow[r] & B \arrow[d] \\ & & b \end{tikzcd}$$

$P_H = \Pr_H[A(y)=1]$

Ideal World

$$\begin{tikzcd} U_{n+2} \arrow[r, "y"] & B \arrow[d] \\ & b \end{tikzcd}$$

$P_I = \Pr[A(y)=1]$

goal: $|P_R - P_I| \leq \delta(n)$ for negligible $\delta(n)$.

Claim: $|P_H - P_I| \leq \varepsilon(n+1)$

by Def of g!

Claim: $|P_R - P_H| \leq \varepsilon(n)$

Proof by Contradiction

If $|P_R - P_H| > \alpha \longrightarrow$ we could get polytime A: dist. $g_1(U_n)$ from U_{n+1} by α.

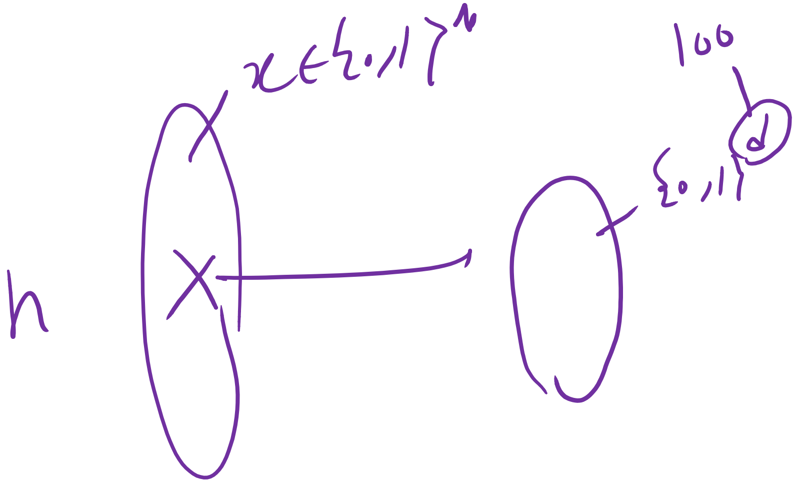
$A(z)$: apply $g_1(z) \to y$ give y to B : output $B(y)$. $\longrightarrow$ A breaks sec of $g_2(\cdot)$

Two main questions:

1. How to get PRGs?

2. Is “indistinguishability-based security” enough in practice?
    a) How to define stronger security notions?
    b) How to achieve them again using PRGs!

Cryptographic Hash Functions

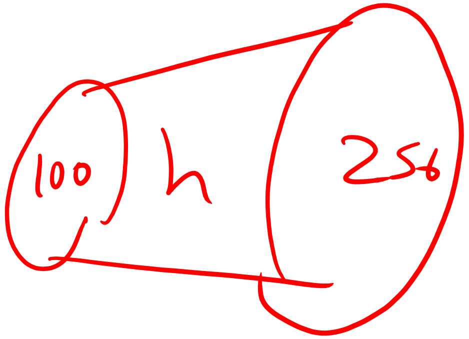


- Two general ways to talk about Hash functions:

1. $h : \{0,1\}^* \to \{0,1\}^d$ for a constant d

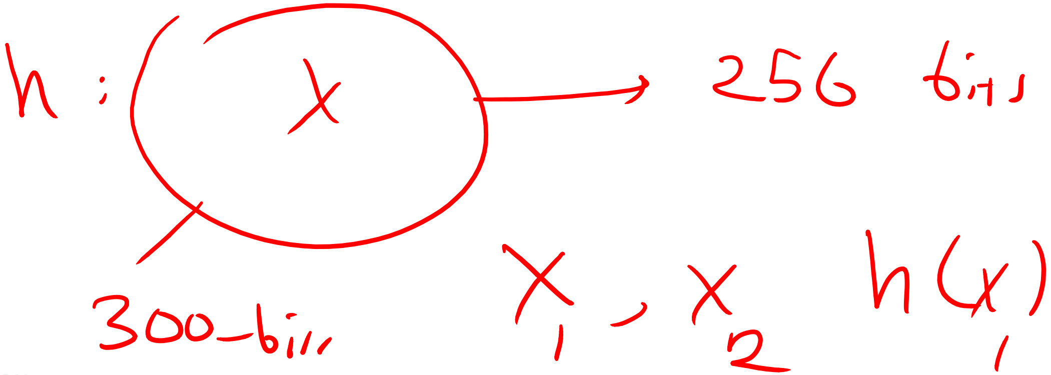
2. $h : \{0,1\}^c \to \{0,1\}^d$ for constants d, c

- The output is called the “message digest”
  - SHA1: 160-bit digest
  - SHA2: 224, 256, 384 or 512 bits
  - SHA3: digest size: arbitrary



Key insight: a “secure” hash shall be unpredictable as it could be (practically like a random function)
In particular, it should be pseudorandom!

- http://www.sha1-online.com/
- https://emn178.github.io/online-tools/sha3_512.html



x_1 , x_2 $h(x_1) \overset{?}{=} h(x_2)$

Cryptology ePrint Archive: Report 2017/190

The first collision for full SHA-1

Marc Stevens and Elie Bursztein and Pierre Karpman and Ange Albertini and Yarik Markov

Abstract: SHA-1 is a widely used 1995 NIST cryptographic hash function standard that was officially deprecated by NIST in 2011 due to fundamental security weaknesses demonstrated in various analyses and theoretical attacks. Despite its deprecation, SHA-1 remains widely used in 2017 for document and TLS certificate signatures, and also in many software such as the GIT versioning system for integrity and backup purposes.

A key reason behind the reluctance of many industry players to replace SHA-1 with a safer alternative is the fact that finding an actual collision has seemed to be impractical for the past eleven years due to the high complexity and computational cost of the attack.

In this paper, we demonstrate that SHA-1 collision attacks have finally become practical by providing the first known instance of a collision.

Furthermore, the prefix of the colliding messages was carefully chosen so that they allow an attacker to forge two distinct PDF documents with the same SHA-1 hash that display different arbitrarily-chosen visual contents.

We were able to find this collision by combining many special cryptanalytic techniques in complex ways and improving upon previous work. In total the computational effort spent is equivalent to $2^{63.1}$ calls to SHA-1's compression function, and took approximately 6,500 CPU years and 100 GPU years. While the computational power spent on this collision is larger than other public cryptanalytic computations, it is still more than 100,000 times faster than a brute force search.

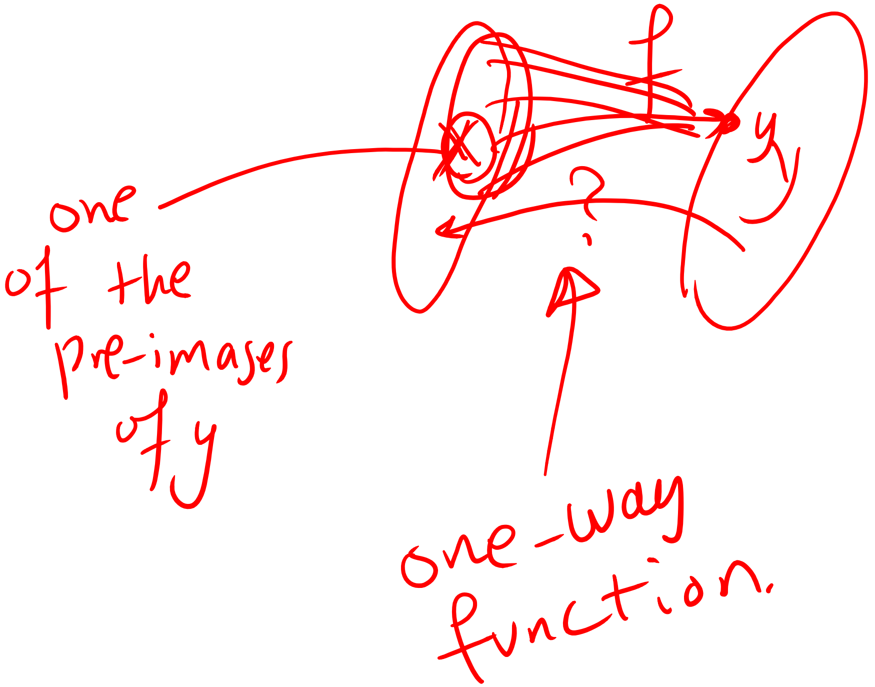
Category / Keywords: public-key cryptography / hash function, cryptanalysis, collision attack, SHA-1, collision example, differential path

Original Publication (with minor differences): IACR-CRYPTO-2017

[Hastad. Impagliazzo Levin Luby] : ∃ OWF → ∃ PRG

Less Practical, but More Robust Constructions

- PRGs based on “one way functions”



$f(x)$ takes $\text{poly}(|x|)$ to compute.

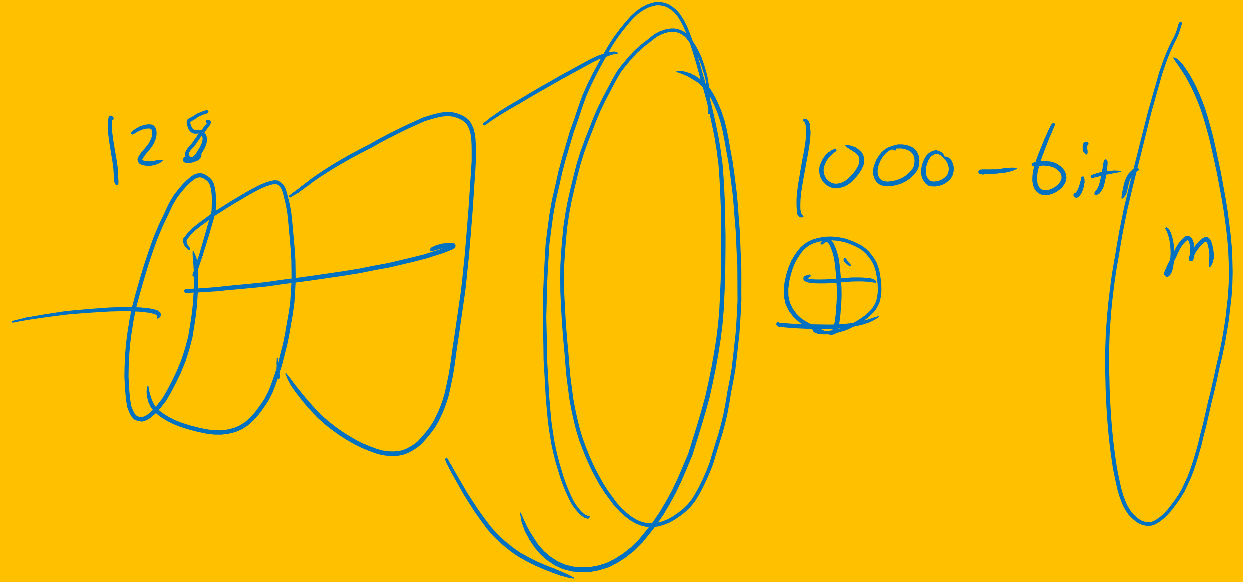
$$\Pr\left[A(y) \to x : f(x)=y\right] \leq \varepsilon(|x|)$$

$y \leftarrow f(U_n)$

g is PRG →

g is one-way.

Two main questions:



1. How to get PRGs?

2. Is “indistinguishability-based security” enough in practice?
   a) How to define stronger security notions?
   b) How to achieve them again using PRGs (or something similar!)

What is wrong with Ind-based definition tailored to *one message* security games?

$m \in \mathcal{M} = \{0,1\}^n$

$k \in \{0,1\}^{n/10}$

$m = m' \longrightarrow \mathrm{Enc}(k,m) = E(k,m')$

"deterministic" encryption

Necessity for *Randomized* Encryption

Enc Scheme is 3 Randomized algs.

(Gen, Enc, Dec).

$$\mathrm{Enc}(K, m ; r) = C \qquad r \in \{0,1\}^n$$

randomness

$$\left\{ \begin{array}{l} c \leftarrow \mathrm{Enc}_k(m) \\ \mathrm{Enc}(k,m) \rightarrow c \\ c \leftarrow \mathrm{Enc}(k,m) \end{array} \right.$$

$$\mathrm{Dec}(k, c) \rightarrow \tilde{m}$$

Completeness $\forall\, k, r :\ \mathrm{Dec}([k, \mathrm{Enc}(k,m;r)) = m]$

Soundness?

$Ch \xrightarrow{k}$ $\forall m, m'$ | A

$Enc(m) \rightarrow c_1$

$Enc(m') \rightarrow c_2$

c_1, c_2

either

$b=0$

$b=1$

$Enc(m) \rightarrow c_1$

$Enc(m') \rightarrow c_2$

$m' \neq m$

A

guess if $b = 0/1$

$(m_0, m_0') \leftarrow A$

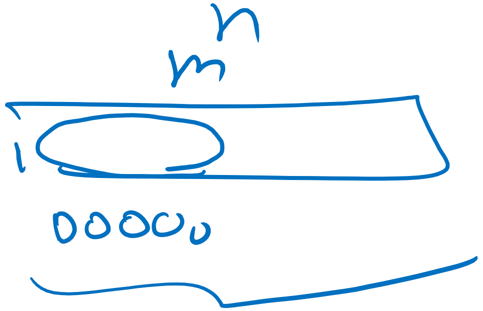
vs

(m_1, m_1')

$b \leftarrow \{0,1\}$

$(Enc(m_b), Enc(m'_b))$

2-mess. sec



insecure

$m \in \mathcal{M}$

$|m| = n$

Good Q: prove that if the scheme is "2-message-secure" under Def then it is "1-message-secure" under our prev. def.

generalize to k-message!

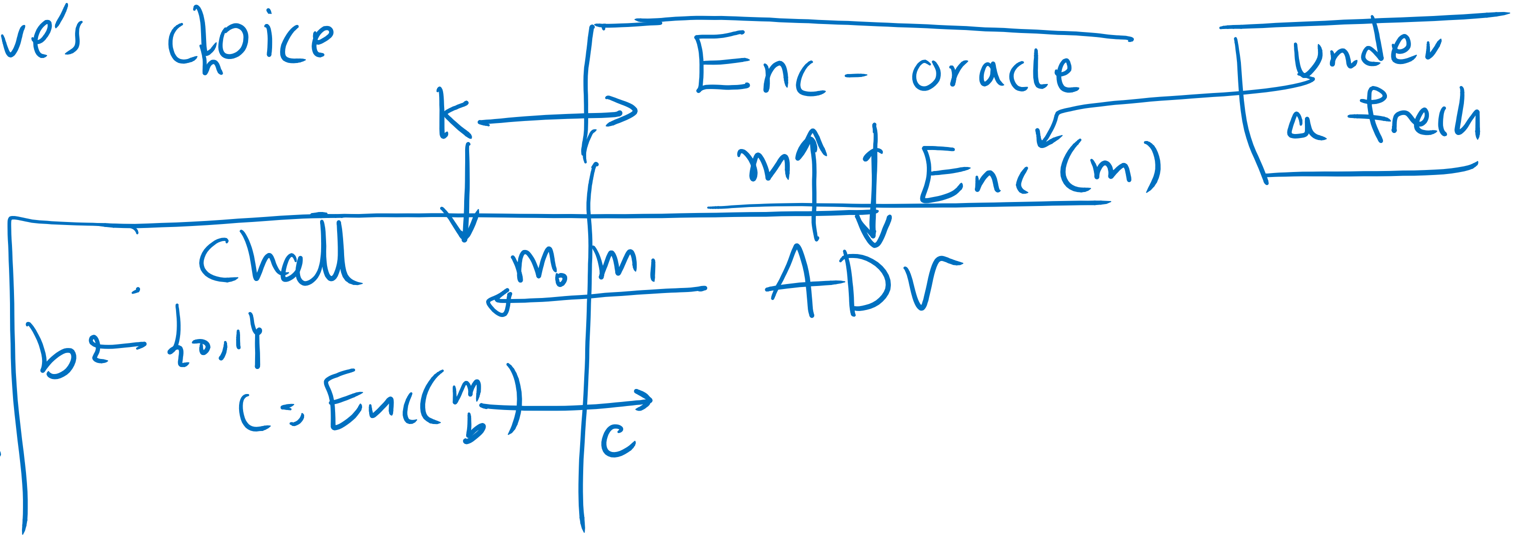
Security against Chosen Plaintext Attacks (CPA Security)

$m_0, m_1 \in |\mathcal{M}|$

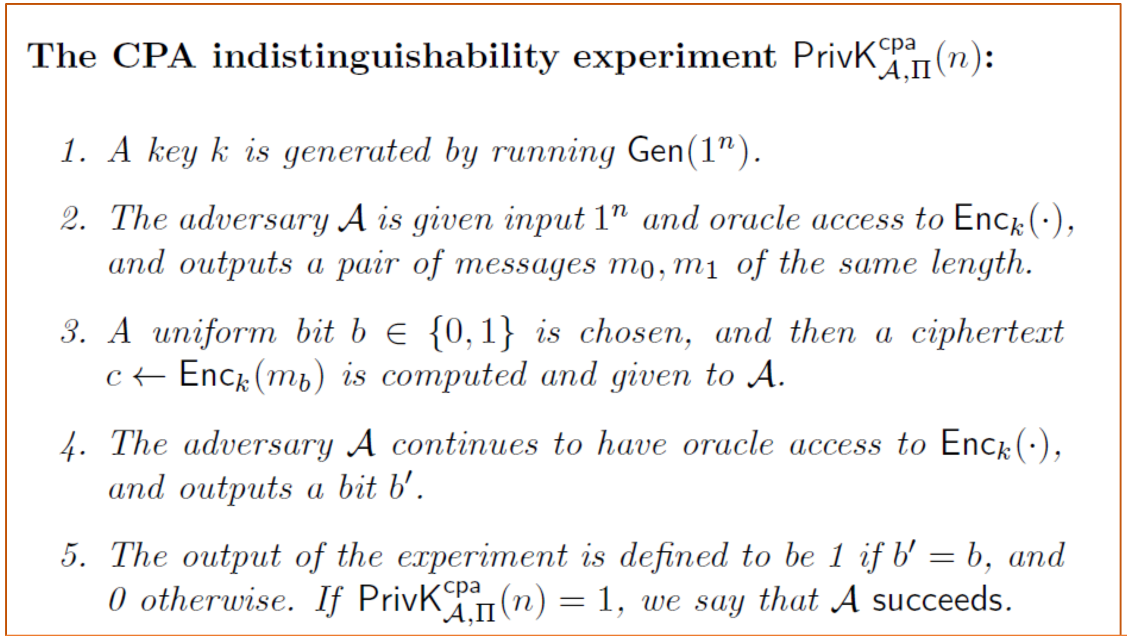
Intuitively: Eve shall not Learn whether Alice enc $\left\{ \begin{matrix} m_0 \\ m_1 \end{matrix} \right.$ even if Eve can request Enc(m) at any time!

Eve's choice

Enc oracle and Enc of chall. are using the SAME key.

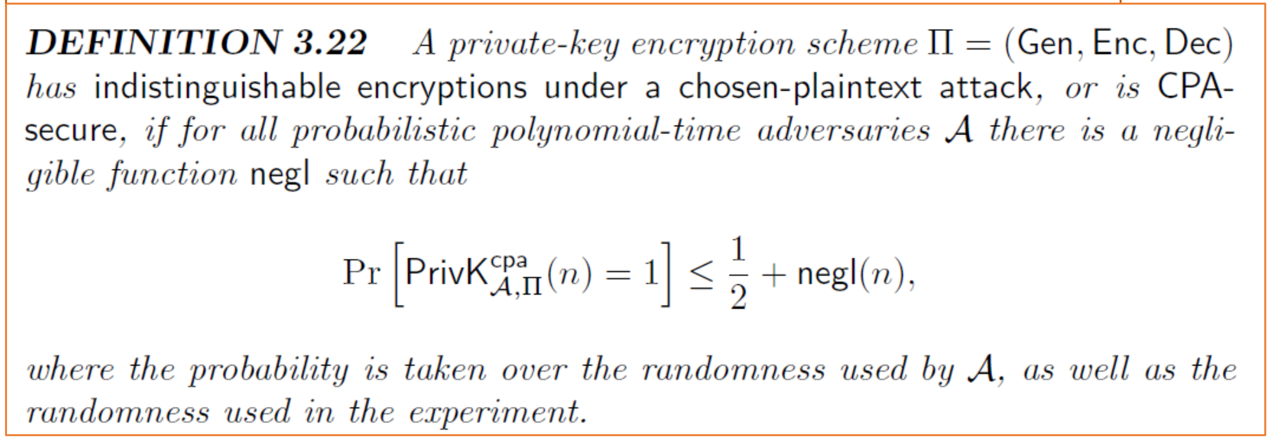


Chosen Plaintext Security



The CPA indistinguishability experiment $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$**:**

1. *A key k is generated by running* $\mathsf{Gen}(1^n)$.
2. *The adversary $\mathcal{A}$ is given input 1^n and oracle access to* $\mathsf{Enc}_k(\cdot)$, *and outputs a pair of messages m_0, m_1 of the same length.*
3. *A uniform bit $b \in \{0,1\}$ is chosen, and then a ciphertext* $c \leftarrow \mathsf{Enc}_k(m_b)$ *is computed and given to $\mathcal{A}$.*
4. *The adversary $\mathcal{A}$ continues to have oracle access to* $\mathsf{Enc}_k(\cdot)$, *and outputs a bit b'.*
5. *The output of the experiment is defined to be 1 if $b' = b$, and 0 otherwise. If* $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1$, *we say that $\mathcal{A}$* succeeds.



DEFINITION 3.22 *A private-key encryption scheme* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *has* indistinguishable encryptions under a chosen-plaintext attack, *or is* CPA-secure, *if for all probabilistic polynomial-time adversaries $\mathcal{A}$ there is a negligible function* negl *such that*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(n),$$

where the probability is taken over the randomness used by $\mathcal{A}$, as well as the randomness used in the experiment.

Next time

1. How to get PRGs?

2. Is “indistinguishability-based security” enough in practice?
   a) How to define stronger security notions?
   b) How to achieve CPA security using PRGs (**or something similar**!)