Introduction to
Cryptography
Lecture 05-07

©Michael Luttenberger

Chair for Foundations of Software Reliability and Theoretical Computer Science
Technische Universität München

2018/10/31, 09:59

Recap: From perfect to comp. secrecy

- ES can only be perfectly secret if $|\mathcal{K}| \geq |\mathcal{M}|$.

- Goal: Apply necessary changes to "perfect secrecy" so that an ES can also be secure when $|\mathcal{K}| < |\mathcal{M}|$.

- Changes:

    - Only consider attacks a thread which run in time polynomial in $\log|\mathcal{K}|$, i.e. the key length.

    $\triangleright$ Not a necessary but a natural restriction:

      super-polynomial attacks become inefficient very fast.

    $\triangleright$ Rules out brute-force computation of $D_c = \{\text{Dec}_k(c) \mid k \in \mathcal{K}\}$ (necessary).

    - Eve may win negligibly better (or worse) than $1/2$.

    $\triangleright$ Necessary as Eve can always guess k instead of m.

- As we assume in general that Eve is at least as powerful as Alice&Bob, also the ES has to run in polynomial time.

- **Definition**: An ES $\mathcal{E} = (\text{Gen}, \text{Enc}, \text{Dec})$ is a PPT-ES if all algorithms run in time polynomial w.r.t. their resp. inputs:

Algorithm	Type	Input	Output
Gen	PPT	1^n	$k \overset{r}{\in} \mathcal{K}_n$ with $|k| \geq n$
Enc	PPT	$k \in \mathcal{K}_n, m \in \mathcal{M}_n$	$c \overset{r}{:=} \text{Enc}_k(m) \in \mathcal{C}_n$
Dec	DPT	$k \in \mathcal{K}_n, c \in \mathcal{C}_n$	$m = \text{Dec}_k(c) \in \mathcal{M}_n$

where $\text{Dec}_k(\text{Enc}_k(m)) = m$ for all $m \in \mathcal{M}, k \in \mathcal{K}$.

The ES is stateless if Enc (Dec) starts from the same initial configuration each time it is called; otherwise it is stateful.

- A stateful ES can remember e.g. a message counter ("q-time pad").

The ES is deterministic if Enc is a DPT-algorithm.

The ES is fixed-length if $\mathcal{M}_n = \Sigma^{l(n)}$ for some function $l(n)$.

- $l(n)$ is usually a polynomial in n as otherwise no PPT-attack could operate with the ES.

- **Example**: The $(\{0,1\}, n)$-one-time pad with $\mathsf{Gen}(1^n)\colon k \overset{u}{\in} \{0,1\}^n$ is a deterministic, stateless, fixed-length PPT-ES.

- In the following, we usually consider ES with

  - $\Sigma = \{0,1\}$,

  - $\mathcal{K}_n = \{0,1\}^n$ where $\mathsf{Gen}(1^n)$ generates $k \overset{u}{\in} \{0,1\}^n$,

  and either

  - $\mathcal{M}_n = \{0,1\}^{l(n)}$ for $l(n) \geq n$, or

  - $\mathcal{M}_n = \{0,1\}^*$, or

  - $\mathcal{M}_n = \left(\{0,1\}^{l(n)}\right)^*$ (e.g. for ES built from block ciphers with block lenght $l(n)$; use a padding function for messages of length not a multiple of $l(n)$.)

  ▷ Statements like "$\mathcal{M}_n = \{0,1\}^*$" or "$\mathcal{M}_n = \left(\{0,1\}^{l(n)}\right)^*$" should be read as "practically arbitrary message length", i.e. at most exponential in n or $l(n)$.

- **Definition**:

  Game INDED for ES $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, attack $\mathcal{A}$, and sec. par. n:

  1 Eve runs $\mathcal{A}(1^n)$ to obtain $m_0, m_1 \in \mathcal{M}_n$ with $|m_0| = |m_1|$.

  2 Alice&Bob generate a random key k by running $\mathsf{Gen}(1^n)$; they then choose $b \stackrel{u}{\in} \{0,1\}$ by tossing a fair coin; finally, they send $c := \stackrel{r}{\mathsf{Enc}_k(m_b)}$ to Eve.

  3 Eve runs $\mathcal{A}(1^n, c)$ to obtain a reply r.

  ▷ Let $\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDED}}(\mathcal{A})$ denote the event that $b = r$.

  A PPT-ES $\mathcal{E}$ has indistinguishable encryptions in the presence of an eavesdropper (is computationally secret) if

  for every PPT-adversary $\mathcal{A}$ its advantage

  $$\varepsilon_{\mathcal{A}}(n) := \left| \Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDED}}(\mathcal{A})\right] - 1/2 \right|$$

  is a negligible function in n.

- Recall: Parameter 1^n only needed to allow $\mathcal{A}$ to run in time polynomial in n.

- Restriction $|m_0| = |m_1|$ necessary as otherwise

  - practical ES would not be comp. secret:

    For efficiency, we usually want $|m| \approx |c|$.

  - in fact, any ES that can encrypt messages of arbitrary length would not be comp. secret.

- **Ex**: Assume we relax the definition of comp. secrecy by allowing $\mathcal{A}$ to output messages of distinct length.

  Show that no PPT-ES $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with $\mathcal{M}_n = \{0, 1\}^*$ can be secure w.r.t. this relaxed definition.

  (Recall, for perfect secrecy we assumed $\mathcal{M}$ to be finite.)

- **Ex**: If a PPT-ES is perfectly secret, then it is also comp. secret.

▷ **Corollary**: As the one-time pad

- $\mathcal{M}_n = \mathcal{C}_n = \mathcal{K}_n = \{0,1\}^n$

- $\mathsf{Gen}(1^n) \colon k \stackrel{u}{\in} \{0,1\}^n$

- $\mathsf{Enc}_k(m) = k \oplus m$

- $\mathsf{Dec}_k(m) = k \oplus m$

is perfectly secret for every n, it is also comp. secret, and every attack $\mathcal{A}$ has zero advantage in winning the game INDED against it.

- As $\mathcal{A}$ runs in time polynomial in n, the message lengths $|m_0|, |m_1|$ are also polynomial in n.

▷ Hence, the input to Enc and Dec is also polynomial in n, and the whole game runs in time polynomial in n.

- Resolving the asymptotics:

▷ "Unpacking" the definition of comp. secrecy we have that

  if $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is comp. secret, then

  $\forall \mathcal{A} \forall c \in \mathbb{N} \exists N_{c,\mathcal{A}} \forall n \geq N_{c,\mathcal{A}}: \left| \Pr\left[\mathsf{Win}^{\mathrm{INDED}}_{n,\mathcal{E}}(\mathcal{A}) \right] - \frac{1}{2} \right| < n^{-c}.$

▷ But nothing is said for when $n < N_{c,\mathcal{A}}$:

  $\mathcal{A}$ might have a much larger advantage than n^{-c}.

▷ To choose the correct value for the security parameter, we need to resolve the asymptotics.

- Assuming that $\mathcal{A}$ is chosen from a finite set "KnownPPTAttacks",

  we then may choose e.g. $c = 3$ and, subsequently,
  $n \geq \max\{1000, N_{3,\mathcal{A}} \mid \mathcal{A} \in \mathsf{KnownPPTAttacks}\}$

  so that all KnownPPTAttacks succeed with prob. at most $1/2 + 10^{-9}$.

- So far, we have only seen that the OTP is also comp. secret.

▷ We still need to find a comp. secret PPT-ES with $|\mathcal{K}| < |\mathcal{M}|$.

- "Cheat": Adapt the OTP to keys which are shorter than the messages by using a "key stretcher" G.

▷ Then use $G(k)$ instead of k in the OTP: $m \oplus G(k)$.

- **Definition** "Key stretcher" G:

  - G stretches bit strings of length n to bit strings of length $l(n) > n$.

    Formally: $\forall x \in \{0,1\}^n : G(x) \in \{0,1\}^{l(n)}$.

  - l is the stretch of G.

  - G can be computed in DPT. (So $l(n)$ is polynomially bounded.)

- **But** when is a key stretcher G "good" enough? When is "$m \oplus G(k)$" a comp. secret ES?

- Can any key stretcher be used?

    - $G(k) = k||k$.

    - $G(k) = k||0$.

    - $G(k) = \mathsf{AES}_k(0^n)||\mathsf{AES}_k(1^n)$ (only for $|k| \in \{128, 192, 256\}$).

- When is a key stretcher G "good"?

- In our setting, Eve is essentially a PPT-algorithm:

    G is "good" if no PPT-algorithm can distinguish (for $n < l$)

    - a truly random string $y \overset{u}{\in} \{0,1\}^l$ (as used in the OTP)

    - from the output $G(x)$ for $x \overset{u}{\in} \{0,1\}^n$ (replacement for y in the OTP).

▷ "Good" key stretchers are called pseudorandom (bit) generators (short: PRG).

    - In some articles: key stretchers "=:" PRGs; good key stretchers "=:" secure PRGs.

- **Definition**: Let G be a key stretcher of stretch $l(n)$.

  G is a pseudorandom generator (PRG) of stretch $l(n)$ if

  every PPT-algorithm $\mathcal{D}$ has negligible advantage

  $$\varepsilon_{\mathcal{D}}(n) := \left| \Pr\left[\mathsf{Win}_{n,G}^{\mathrm{INDPRG}}(\mathcal{D}) \right] - 1/2 \right|$$

  in the game INDPRG:

  1. Alice&Bob toss a fair coin $b \overset{u}{\in} \{0,1\}$;
     if $b = 0$, then $y \overset{u}{\in} \{0,1\}^{l(n)}$;
     if $b = 1$, then $x \overset{u}{\in} \{0,1\}^n$ and $y := G(x)$;
     they pass y to Eve;

  2. Eve runs $\mathcal{D}(1^n, y)$ to obtain the reply r.

  ▷ Let $\mathsf{Win}_{n,G}^{\mathrm{INDPRG}}(\mathcal{D})$ denote the event $r = b$.

- **Example**: Assuming that AES is a "secure block" cipher,

  $G(k) = \mathsf{AES}_k(0^{128})||\mathsf{AES}_k(1^{128})$ is a PRG. (later)

  - The Fortuna prg [1] is based on this assumption.

- **Ex**: Is the following key stretcher with $l(n) = n + 1$ a PRG?

  $G(x_1 x_2 \ldots x_n) = x_1 x_2 \ldots x_n x_2$

- **Ex**: Let $f \colon \{0,1\}^* \to \{0,1\}$ be any DPT-computable function.

  Show that $G_f(x) = x||f(x)$ is no PRG.

- **Ex**: Some books use a slightly different definition where

  $$\left| \Pr_{x \in \{0,1\}^n} [\mathcal{D}(1^n, G(x)) = 1] - \Pr_{y \in \{0,1\}^{l(n)}} [\mathcal{D}(1^n, y) = 1] \right|$$

  has to be negligible. Show that this is equivalent to our definition here.

- Usually, we want a PRG whose stretch $l(n)$ is much larger than n.

  - E.g. let G be a PRG of stretch $l(n) = 2n$.

▷ **Ex**: Then G can only output a negligible fraction of $\{0,1\}^{2n}$.

  In particular, there is an exponential time distinguisher $\mathcal{D}$ with:

  $$\left| \Pr_{x \overset{u}{\in} \{0,1\}^n}[\mathcal{D}(1^n, G(x)) = 1] - \Pr_{y \overset{u}{\in} \{0,1\}^{2n}}[\mathcal{D}(1^n, y) = 1] \right| \geq 1 - 2^{-n}.$$

▷ So, the output of G is far from uniformly distributed over $\{0,1\}^{2n}$.

  - Hence the term pseudorandom.

- **Ex**: Determine the success prob. of the following $\mathcal{D}$ for $l(n) \geq 2n$:

  - Input: $y \in \{0,1\}^{l(n)}$ and 1^n.

  - Generate $x' \overset{u}{\in} \{0,1\}^n$ and compute $y' = G(x')$.

  - Return 1 if $y = y'$; else return 0.

  (Recall why comp. secrecy allows a negligible advantage.)

- We can also reformulate the game $\textsc{IndPRG}$ using the oracles

$\mathcal{O}_0$ ("perfect world")	$\mathcal{O}_1$ ("real world")
Init: 1^n	Init: 1^n
Query: outputs $y \overset{u}{\in} \{0,1\}^{l(n)}$	Query: outputs $G(x)$ with $x \overset{u}{\in} \{0,1\}^n$.

$\triangleright$ In the game, Alice&Bob pass $\mathcal{O}_b$ in a black box $\mathcal{O}$ to $\mathcal{D}$ – where we now require that $\mathcal{D}^{\mathcal{O}}(1^n)$ makes at most one oracle query.

$\triangleright$ G is a PRG, if Eve cannot distinguish (with a single query) the "real world $\mathcal{O}_1$" from the "perfect world $\mathcal{O}_0$" – except for negligible prob.

- **Ex**: Analyze the advantage of the following distinguisher $\mathcal{D}$ which is allowed to query the oracle q times:

    - Init: $L = \emptyset$

    - Repeat: query $\mathcal{O}$ to obtain y; if $y \in L$, stop and output 1; if $|L| > q$, stop and output 0; else $L := L \cup \{y\}$.

- **Definition**:

  Let $G(\cdot)$ be a PRG of polynomial stretch $l(\cdot)$.

  The pseudorandom one-time pad w.r.t. $G(\cdot)$ (G-prOTP) is defined by:

  - $\mathsf{Gen}(1^n)$ outputs a key $k \overset{u}{\in} \mathcal{K}_n = \{0,1\}^n$ uniformly at random.

  - $\mathsf{Enc}_k(m) := m \oplus G(k)$ for every $m \in \mathcal{M}_n = \{0,1\}^{l(n)}$.

  - $\mathsf{Dec}_k(c) := c \oplus G(k)$ for every $c \in \mathcal{C}_n = \{0,1\}^{l(n)}$.

    (Note that $\oplus \equiv \ominus$ for $\Sigma = \{0,1\}$.)

- **Theorem**:

  The G-prOTP is a comp. secret ES of fixed-length $l(\cdot)$ if G is a PRG of stretch $l(\cdot)$.

- Let $\mathcal{A}$ be any PPT-attack on the pseudorandom one-time pad in the "PPT-eavesdropping" game INDED.

- We construct from $\mathcal{A}$ a distinguisher $\mathcal{D}$ for G which succeeds with roughly the same prob. in INDPRG as $\mathcal{A}$ does in INDED.

- ▷ This will allow us to conclude that $\mathcal{A}$ can only have negligible advantage as G is a PRG and thus $\mathcal{D}$ can only win with negligible advantage.

- Construction of $\mathcal{D}$:

  $\mathcal{D}$ takes the place of Alice and Bob, and simulates the game INDED for n using Eve's algorithm $\mathcal{A}$ as a subprocedure.

  Except that $\mathcal{D}$ does not run Gen

  but instead assumes that its input y is the output of $G(k)$ in order to simulate $\mathsf{Enc}_k(m)$ by means of $c = m \oplus y$.

Alice&Bob	$\mathcal{D}$	sub: $\mathcal{A}$
$b \overset{u}{\in} \{0,1\}$		
if $b = 0$: $y \overset{u}{\in} \{0,1\}^{l(n)}$		
if $b = 1$: $x \overset{u}{\in} \{0,1\}^n, y = G(x)$		
pass y to $\mathcal{D}$		
	run $\mathcal{A}(1^n)$	
		return $m_0, m_1 \in \{0,1\}^{l(n)}$
	$b' \overset{u}{\in} \{0,1\}$	
	compute $c = m_{b'} \oplus y$	
	run $\mathcal{A}(1^n, c)$	
		return r'
	return $r := (r' \overset{?}{=} b')$	

- $\mathcal{D}$ wins iff $r = b$.

- Running time $T_{\mathcal{D}}(n)$ of $\mathcal{D}$ given by running time $T_{\mathcal{A}}(n)$ of $\mathcal{A}$ plus time needed for simulating game INDED ($\mathcal{O}(n)$).

- Case $b = 0$: Inserting $b = 0$

Alice&Bob	$\mathcal{D}$	sub: $\mathcal{A}$
~~$b \stackrel{u}{\in} \{0,1\}$~~ $b := 0$ ~~if $b = 0$:~~ $y \stackrel{u}{\in} \{0,1\}^{l(n)}$ ~~if $b = 1$: $x \stackrel{u}{\in} \{0,1\}^n, y = G(x)$~~ pass y to $\mathcal{D}$		
	run $\mathcal{A}(1^n)$	
		return $m_0, m_1 \in \{0,1\}^{l(n)}$
	$b' \stackrel{u}{\in} \{0,1\}$ compute $c = m_{b'} \oplus y$ run $\mathcal{A}(1^n, c)$	
		return r'
	return $r := (r' \stackrel{?}{=} b')$	

- $\mathcal{D}$ wins iff $r = 0$ iff $r' \neq b'$

- Case $b = 0$: From $\mathcal{A}$'s perspective, it is playing vs. the OTP

Alice&Bob (&$\mathcal{D}$)	sub: $\mathcal{A}$
$b := 0$	
$y \overset{u}{\in} \{0,1\}^{l(n)}$	
~~pass y to $\mathcal{D}$~~	
run $\mathcal{A}(1^n)$	
	return $m_0, m_1 \in \{0,1\}^{l(n)}$
$b' \overset{u}{\in} \{0,1\}$	
compute $c = m_{b'} \oplus y$	
run $\mathcal{A}(1^n, c)$	
	return r'
$r := (r' \overset{?}{=} b')$	

- Hence, $\mathcal{D}$ wins vs. G iff $r = 0$ iff $r' \neq b'$ iff $\mathcal{A}$ loses vs. the OPT.

▷ $\mathrm{Pr}_{b=0}\big[\mathsf{Win}_{n,G}^{\textsc{IndPRG}}\mathcal{D}\big] = 1 - \mathrm{Pr}\big[\mathsf{Win}_{n,\mathsf{OTP}}^{\textsc{IndEd}}(\mathcal{A})\big] = 1/2.$

- Case $b = 1$: Inserting $b = 1$

Alice&Bob	$\mathcal{D}$	sub: $\mathcal{A}$
$\overset{u}{b \in \{0,1\}}$ $b := 1$		
if $b = 0$: $y \overset{u}{\in} \{0,1\}^{l(n)}$		
if $b = 1$: $x \overset{u}{\in} \{0,1\}^n, y = G(x)$		
pass y to $\mathcal{D}$		
	run $\mathcal{A}(1^n)$	
		return $m_0, m_1 \in \{0,1\}^{l(n)}$
	$b' \overset{u}{\in} \{0,1\}$	
	compute $c = m_{b'} \oplus y$	
	run $\mathcal{A}(1^n, c)$	
		return r'
	return $r := (r' \overset{?}{=} b')$	

- $\mathcal{D}$ wins iff $r = 1$ iff $r' = b'$

- Case $b = 1$: From $\mathcal{A}$'s perspective, it is playing vs. G-prOTP

Alice&Bob (&$\mathcal{D}$)	sub: $\mathcal{A}$
$b := 1$	
$x \stackrel{u}{\in} \{0,1\}^n$	
~~pass y to $\mathcal{D}$~~	
run $\mathcal{A}(1^n)$	
	return $m_0, m_1 \in \{0,1\}^{l(n)}$
$b' \stackrel{u}{\in} \{0,1\}$	
compute $c = m_{b'} \oplus G(x)$	
run $\mathcal{A}(1^n, c)$	
	return r'
$r := (r' \stackrel{?}{=} b')$	

- Hence, $\mathcal{D}$ wins vs. G iff $r = 1$ iff $r' = b'$ iff $\mathcal{A}$ wins vs. G-prOPT.
- ▷ $\mathrm{Pr}_{b=1}\big[\mathsf{Win}_{n,G}^{\mathrm{INDPRG}}\mathcal{D}\big] = \mathrm{Pr}\big[\mathsf{Win}_{n,G\text{-prOTP}}^{\mathrm{INDED}}(\mathcal{A})\big].$

- As $b \stackrel{u}{\in} \{0,1\}$, both cases occur with prob. $1/2$, so:

$$\Pr\left[\mathrm{Win}_{n,G}^{\mathrm{INDPRG}}(\mathcal{D})\right] = \tfrac{1}{4} + \tfrac{1}{2}\Pr\left[\mathrm{Win}_{n,G\text{-prOTP}}^{\mathrm{INDED}}(\mathcal{A})\right]$$

- As G is assumed to be a PRG,

$$
\begin{aligned}
\varepsilon_{\mathcal{D}}(n) &= \left|\Pr\left[\mathrm{Win}_{n,G}^{\mathrm{INDPRG}}(\mathcal{D})\right] - \tfrac{1}{2}\right| \\
&= \tfrac{1}{2}\left|\Pr\left[\mathrm{Win}_{n,G\text{-prOTP}}^{\mathrm{INDED}}(\mathcal{A})\right] - \tfrac{1}{2}\right| = \tfrac{1}{2}\varepsilon_{\mathcal{A}}(n)
\end{aligned}
$$

  has to be negligible.

- As $\mathcal{A}$ was chosen arbitrarily, the G-prOTP is comp. secret.

▷ In other words:

  Given a $(T_{\mathcal{A}}(n), \varepsilon_{\mathcal{A}}(n))$-attack $\mathcal{A}$ on the pseudorandom one-time pad in the game INDED

  we can build a $(T_{\mathcal{D}}(n), \varepsilon_{\mathcal{D}}(n))$-distinguisher $\mathcal{D}$ for G

  where $T_{\mathcal{D}}(n) = T_{\mathcal{A}}(n) + \mathcal{O}(n)$ and $\varepsilon_{\mathcal{A}}(n) = 2\varepsilon_{\mathcal{D}}(n)$.   □

- So, our definition of PRG is sufficient for the existence of comp. secret PPT-ES with $|\mathcal{K}| < |\mathcal{M}|$.

- Later: Our definition is also "the right one"

  as PRGs exist if comp. secret ES exist!

▷ PRGs can be built from problems which are sufficiently hard on the average, e.g.:

  - Factorizing an integer N which is the product of two unknown random primes p, q with $p, q \in [2^{n-1}, 2^n)$. ("BBS PRG")

  - Given a prime $p \in [2^{n-1}, 2^n)$, a generator g of $\mathbb{Z}_p$, and a random $y \overset{u}{\in} \mathbb{Z}_p - \{0\}$, find the unique $x \in \mathbb{Z}_p$ with $g^x \pmod{p} = y$. (BM PRG)

▷ These PRGs have in fact variable stretch.

- Computational secrecy "=" perfect secrecy + necessary changes to obtain a reasonable definition of security for (PPT) ES with $|\mathcal{K}| < |\mathcal{M}|$.

- Pseudorandom generator (PRG) "=" secure key stretchers

  No PPT-adversary can do better than guessing (+ negligible advantage) when he has to distinguish

    - the "perfect world" where $y \overset{u}{\in} \{0,1\}^s$ from

    - the "real world" where $y := G(x)$ with $x \overset{u}{\in} \{0,1\}^n$ and $n < s$.

- Pseudorandom one-time pad (PROTP) "=" $\mathsf{Enc}_k(m) = m \oplus G(k)$

  Is computationally secret iff G is a PRG.

- More practical PRGs: user can choose the stretch s.

- **Definition**: A PRG G of variable stretch not only takes a random $\overset{u}{\in} \{0,1\}^n$ as input, but also a stretch parameter 1^s such that

  ❶ $G(x, 1^s) \in \{0,1\}^s$

  ❷ G is DPT-computable w.r.t. $n + s$.

  ❸ For every x, and $s < s'$, $G(x, 1^s)$ is a prefix of $G(x, 1^{s'})$.

    ("prefix property")

  ❹ G is a "normal" PRG when fixing its stretch, i.e.:

    For every polynomial $l(n)$, $G_{l(\cdot)}(x) := G(x, 1^{l(|x|)})$ is a PRG of fixed stretch $l(n)$.

- **Remarks**:

- Argument 1^s gives G enough time to output s bits.

- Prefix property: Stretching the key further means that only new bits get appended.

- A variable stretch PRG is sometimes called a stream cipher.

  See ECRYPT stream cipher project for a list of currently recommended stream ciphers.

- Later: The definition of secure block cipher F will essentially say

$$G(k, 1^{sl}) = F_k(x_1)||F_k(x_2)|| \ldots ||F_k(x_s)$$

  is a vlPRG for any enumeration $(x_i)_{i \in \{0,1\}^l}$ of $\{0,1\}^l$.

- **Example**: Assuming that AES is a "secure" block cipher,

  $G(k, 1^{l \cdot s}) = \mathsf{AES}_k(\lfloor 0 \rceil)||\mathsf{AES}_k(\lfloor 1 \rceil)|| \ldots ||\mathsf{AES}_k(\lfloor s \rceil)$

  is a PRG of variable length (w.r.t. some concrete bound).

  - This is what Fortuna [1] essentially does to generate its actual input.

  - A much larger part of Fortuna is concerned with generating the actual seed/key k from entropy sources like internal statistics of the operating system, and refreshing the key k after every output.

- **Example**: Blum-Micali (BM) PRG:

  1. Generate from the random x: a prime p, a generator g of $\mathbb{Z}_p$, and a random $y \in \mathbb{Z}_p - \{0\}$.

  2. Do s-times: output $(y < \frac{p-1}{2} \, ? \, 1 \, : \, 0)$; set $y := (g^y \bmod p)$.

▷ **Ex**: Linear congruential generator

Let $m \in \mathbb{N}$ and $a, c \in \mathbb{Z}_m$.

- For simplicity, let $m = 2^n$ so that we may identify $\mathbb{Z}_m$ with $\{0,1\}^n$.

For $x \in \{0,1\}^n$, let $f(x) = (a \cdot x + c) \bmod m$, and
$G(x, 1^{n \cdot s}) = f(x) || f(f(x)) || \ldots || f^s(x)$.

Show that this is a not a PRG (in the sense of cryptography).

- **Ex**: For G a PRG of variable stretch, we may define a "variable-length G-prOTP":

    - $\mathsf{Gen}(1^n) : k \overset{u}{\in} \{0,1\}^n$.

    - $\mathsf{Enc}_k(m) = G(k, 1^{|m|}) \oplus m$

    - $\mathsf{Dec}_k(c) = G(k, 1^{|c|}) \oplus c$

  with $\mathcal{M}_n = \{0,1\}^*$.

  Show that $\mathcal{E}$ is comp. secret if G is a PRG of variable length.

  Hint: Let $\mathcal{A}$ be a PPT-attack on $\mathcal{E}$ in the game INDED. Denote by $T_{\mathcal{A}}$ the running time of $\mathcal{A}$. As before, construct from $\mathcal{A}$ a distinguisher $\mathcal{D}$ for $G_{T_{\mathcal{A}}(\cdot)}$.

  Make use of the requirement that $G(x, 1^s)$ is a prefix of $G(x, 1^{s'})$ for $s < s'$.

- **Definition**: Game INDMULTED

  ① Eve runs $\mathcal{A}(1^n)$ to obtain message sequences $\vec{m}_0, \vec{m}_1$ with $\vec{m}_b = (m_b^{(1)}, \ldots, m_b^{(q)})$, and $\left| m_0^{(i)} \right| = \left| m_1^{(i)} \right|$, and $m_j^{(i)} \in \mathcal{M}_n$.

  ② Alice&Bob generate a random key k by running $\mathsf{Gen}(1^n)$; they then choose $b \overset{u}{\in} \{0, 1\}$ by tossing a fair coin; they compute $c^{(i)} = \mathsf{Enc}_k(m^{(i)})$ from left $i = 1$ to right $i = q$; finally, they send $\vec{c} = (c^{(1)}, \ldots, c^{(q)})$ to Eve.

  ③ Eve runs $\mathcal{A}(1^n, \vec{c})$ to obtain a reply r.

  ▷ Let $\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDMULTED}}(\mathcal{A})$ denote the event that $b = r$.

  A PPT-ES $\mathcal{E}$ has indistinguishable mulitple encryptions in the presence of an eavesdropper if for every PPT-adversary $\mathcal{A}$ its advantage

  $$\varepsilon_{\mathcal{A}}(n) := \left| \Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDMULTED}}(\mathcal{A}) \right] - 1/2 \right|$$

  is a negligible function in n.

- **Theorem**: No stateless and deterministic ES has indistinguishable mulitple encryptions in the presence of an eavesdropper.

- Proof: **Ex**.

▷ ECB mode is not secure for multiple encryptions.

- **Ex***: Turn the "variable-length G-prOTP" into a stateful ES which has indistinguishable mulitple encryptions in the presence of an eavesdropper.

  - Make use of the "prefix property" of G, and remember how "far" the key has already been stretched.

  - Extend the ciphertext accordingly.

  - See also the exercise on the "q-time pad".

Adapt the proof of the security of the G-prOTP.

- In the security definitions discussed so far Eve could
  - influence the messages encrypted by Alice&Bob
  - eavesdrop the ciphertexts exchanged over the public channel.
- But in the real world, Eve might also have limited ways to ask Alice&Bob to
  - encrypt messages chosen by her

    (chosen-plaintext attack (CPA))

    or even
  - encrypt and decrypt data chosen by her

    (chosen-ciphertext attack (CCA))

  as motivated by the following examples.

- Historical example for CPA [2]:

  "In May 1942, US Navy cryptanalysts had discovered that Japan was planning an attack on Midway island[...]. They had learned this by intercepting a communication containing the ciphertext fragment "AF" that they believed corresponded to the plaintext "Midway islands". Unfortunately, their attempts to convince Washington planners that this was indeed the case were futile; the general belief was that Midway could not possibly be the target. The Navy cryptanalysts then [...] instructed the US forces at Midway to send a plaintext message that their freshwater supplies were low. The Japanese [...] immediately reported [...] that "AF" was low on water."

- Public-key encryption: Eve can always encrypt herself as everybody knows the public encryption key.

- Example of CCA: Needham-Schroeder protocol, 1978

  A variant of this protocol describes how Alice and Bob can setup a public-key communication using a trusted key distribution center which is used to store their most recent public keys.

  In its original version the protocol is vulnerable to a man-in-the-middle attack first published by Gavin Lowe in 1995:

  For this attack, Eve waits until Alice tries to setup communication with her; Eve then immediately also initiates communication with Bob, and intertwines the two protocol executions such that Bob in the end mistakes Eve for Alice.

  One essential step of this attack consists of Eve using Alice as decryption oracle in order to decrypt a message sent by Bob to Alice. (4th and 5th in the attack described on wikipedia)

- RSAES-PKCS1: broken by a CCA-attack.

- To model these kinds of attack, we will extend the definition of comp. secrecy:

  We give Eve also oracle access to the ES instantiated on the secret key k:

  - Recall: Oracel "=" black box; she may only query the black box and analyze the replies.

  - oracle access to Enc_k allows Eve to obtain the encryption of "Midway is low on water".

  - oracle access to $\text{Dec}_{\text{dk}_{\text{Alice}}}$ allows Eve impersonate Alice in the Needham-Schroeder protocol.

- By doing so, we arrive at the definitions of CPA- and CCA-security.

- **Definition**: Game $\textsc{IndCPA}$:

  1 <u>Alice&Bob</u> generate a random key $k := \mathsf{Gen}(1^n)$ and give Eve's attack $\mathcal{A}$ oracle access to Enc_k.

  2 Eve runs $\mathcal{A}^{\mathsf{c}_k}(1^n)$ to obtain two sequences $\vec{m}_0, \vec{m}_1$ with $\vec{m}_b = (m_b^{(1)}, \ldots, m_b^{(q)})$, and $\left| m_0^{(j)} \right| = \left| m_1^{(j)} \right|$, and $m_b^{(i)} \in \mathcal{M}_n$.

  3 Alice&Bob choose $b \in \{0,1\}$ by tossing a fair coin, compute $c^{(i)} = \mathsf{Enc}_k(m_b^{(i)})$ from left $i = 1$ to right $i = q$, and send $\vec{c} = (c^{(1)}, \ldots, c^{(q)})$ to Eve.

  4 Eve runs $\mathcal{A}^{\mathsf{Enc}_k}(1^n, \vec{c})$ to obtain her reply $r \in \{0,1\}$.

  ▷ Let $\mathsf{Win}_{n,\mathcal{E}}^{\textsc{IndCPA}}$ be the event that $b = r$.

An ES $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has indistinguishable (multiple) encryptions under a chosen-plaintext attack (is CPA-secure) if every PPT-attacks $\mathcal{A}$ has only negligible advantage $\varepsilon_{\mathcal{A}}(n) = \left| \Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\textsc{IndCPA}}(\mathcal{A}) \right] - \frac{1}{2} \right|$.

- **Definition**: Game INDCCA:

  1 Alice&Bob generate a random key $k := \mathsf{Gen}(1^n)$ and give Eve's attack $\mathcal{A}$ oracle access to Enc_k and Dec_k.

  2 Eve runs $\mathcal{A}^{\mathsf{Enc}_k, \mathsf{Dec}_k}(1^n)$ to obtain two sequences $\vec{m}_0, \vec{m}_1$ with $\vec{m}_b = (m_b^{(1)}, \ldots, m_b^{(q)})$, and $\left| m_0^{(j)} \right| = \left| m_1^{(j)} \right|$, and $m_b^{(i)} \in \mathcal{M}_n$.

  3 Alice&Bob choose $b \in \{0, 1\}$ by tossing a fair coin, compute $c^{(i)} = \mathsf{Enc}_k(m_b^{(i)})$ from left $i = 1$ to right $i = q$, and send $\vec{c} = (c^{(1)}, \ldots, c^{(q)})$ to Eve.

  4 Eve runs $\mathcal{A}^{\mathsf{Enc}_k, \mathsf{Dec}_k}(1^n, \vec{c})$ to obtain her reply $r \in \{0, 1\}$ where $\mathcal{A}^{\mathsf{Enc}_k, \mathsf{Dec}_k}$ is not allowed to query Dec_k for any $c^{(i)}$. 📝

  ▷ Let $\mathsf{Win}_{n, \mathcal{E}}^{\mathrm{INDCCA}}$ be the event that $b = r$.

An ES $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has indistinguishable (multiple) encryptions under an adaptive chosen-ciphertext attack (is CCA-secure) if every PPT-attacks $\mathcal{A}$ has only negligible advantage $\varepsilon_{\mathcal{A}}(n) = \left| \Pr\left[\mathsf{Win}_{n, \mathcal{E}}^{\mathrm{INDCCA}}(\mathcal{A}) \right] - \frac{1}{2} \right|$.

- **Lemma**: Let $\mathcal{E}$ be an ES. Then:

  If $\mathcal{E}$ is CCA-secure, $\mathcal{E}$ is CPA-secure.

  If $\mathcal{E}$ is CPA-secure, $\mathcal{E}$ is comp. secret w.r.t. multiple encryptions using the same key.

- ▷ Proof: **Ex**.

- **Lemma**:

  No stateless and deterministic ES can be CPA-secure (or CCA-secure).

- ▷ Proof: **Ex** (recall the OTP).

- **Example**: CPA-security implies security against plaintext recovery

- Consider the experiment

$$k \stackrel{r}{:=} \mathsf{Gen}(1^n);$$
$$m \stackrel{u}{\in} \{0,1\}^{l(n)};$$
$$c \stackrel{r}{:=} \mathsf{Enc}_k(m);$$
$$m' \stackrel{r}{:=} \mathcal{B}^{\mathsf{Enc}_k}(1^n, c);$$
$$\text{WIN: } m' = m$$

  where $\{0,1\}^{l(n)} \subseteq \mathcal{M}_n$.

▷ **Ex***: Assume that the ES is CPA-secure (with $l(n) \geq n$)

  Show that then every PPT-algorithm $\mathcal{B}$ can only succeed with negl. prob. in n in above experiment.

- Define $\text{CPA}^{\text{single}}$-security resp. $\text{CCA}^{\text{single}}$-security by restricting $\mathcal{A}$ to choose a single message pair $m_0, m_1 \in \mathcal{M}_n$ with $|m_0| = |m_1|$ instead of two message sequences.

- The following theorem allows us to restrict our analysis (at least for asymptotic bounds) to the case of a single encryption when studying CPA or CCA attacks:

- **Theorem**:

  An ES is $\text{CPA}^{\text{single}}$-secure iff it is CPA-secure.

  An ES is $\text{CCA}^{\text{single}}$-secure iff it is CCA-secure.

- ▷ "$\Leftarrow$": obvious.

- ▷ "$\Rightarrow$": (Main idea) We can restrict a "multiple encryptions"-attack to a "single-encryption"-attack as we can use the oracle access to Enc_k to simulate the remaining encryptions.

Alice&Bob	$\mathcal{A}_{\text{single}}$	$\mathcal{A}_{\text{mult}}$
generate $k := \text{Gen}(1^n)$		
run $\mathcal{A}_{\text{single}}^{\text{Enc}_k}(1^n)$		
	run $\mathcal{A}_{\text{mult}}^{\text{Enc}_k}(1^n)$	
		return $\vec{m}_0, \vec{m}_1$
	choose $s \overset{u}{\in} [q]$	
	for $i = 1$ to $i = s - 1$:	
	$\quad c^{(i)} := \text{Enc}_k(m_1^{(i)})$	
	return $(m_0^{(s)}, m_1^{(s)})$	
choose $b \overset{u}{\in} \{0, 1\}$		
$c := \text{Enc}_k(m_b^{(s)})$		
run $\mathcal{A}_{\text{single}}^{\text{Enc}_k}(1^n, c)$		
	set $c^{(s)} := c$	
	for $i = s + 1$ to $s = q$:	
	$\quad c^{(i)} := \text{Enc}_k(m_0^{(i)})$	
	run $\mathcal{A}_{\text{mult}}^{\text{Enc}_k}(1^n, \vec{c})$	
		return r
	return r	

- **Remark**: For simplicity, $\mathcal{A}_{\text{mult}}$ uses always $q := q(n)$ messages in $\vec{m}_i$.

- Assume $q = 3$, then $\mathcal{A}_{\mathsf{mult}}(1^n, \vec{c})$ will get an encryption of

  - $\vec{m}^1 := m_0^{(1)} m_0^{(2)} m_0^{(3)}$ if $s = 1 \wedge b = 0$.

  - $\vec{m}^2 := m_1^{(1)} m_0^{(2)} m_0^{(3)}$ if $s = 1 \wedge b = 1$ or $s = 2 \wedge b = 0$

  - $\vec{m}^3 := m_1^{(1)} m_1^{(2)} m_0^{(3)}$ if $s = 2 \wedge b = 1$ or $s = 3 \wedge b = 0$

  - $\vec{m}^4 := m_1^{(1)} m_1^{(2)} m_1^{(3)}$ if $s = 3 \wedge b = 1$.

- Only for $s = 1 \wedge b = 0$ and $s = 3 \wedge b = 1$, $\mathcal{A}_{\mathsf{mult}}$ will get an encryption of the expected messages – these are the cases we actually have to analyze.

- The remaining cases are actually not relevant for the success probability of $\mathcal{A}_{\mathsf{mult}}$ BUT they simplify the analysis as e.g.

$$\Pr_{s=1, b=1}\left[\mathcal{A}_{\mathsf{mult}}^{\mathsf{Enc}_k}(1, \vec{c}) = b\right] = 1 - \Pr_{s=2, b=0}\left[\mathcal{A}_{\mathsf{mult}}^{\mathsf{Enc}_k}(1, \vec{c}) = b\right]$$

as c is here an encryption of $\vec{m}^2$, and $\mathcal{A}_{\mathsf{mult}}^{\mathsf{Enc}_k}$ is an algorithm.

▷ For general $q = q(n)$ messages per $\vec{m}_i$:

- $\Pr_{b=0,s=q}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] = \Pr_{b=0}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{mult}})\right],$

- $\Pr_{b=1,s=1}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] = \Pr_{b=1}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{mult}})\right],$ and

- $\Pr_{b=0,s=i}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] + \Pr_{b=1,s=i+1}\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] = 1.$

▷ Summing up over all cases for b and s yields:

$$\Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] = \frac{1}{q}\Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{mult}})\right] + \frac{q-1}{2q}$$

▷ and subsequently:

$$\left|\Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{mult}})\right] - 1/2\right| \leq q\left|\Pr\left[\mathsf{Win}_{n,\mathcal{E}}^{\mathrm{INDCPA}}(\mathcal{A}_{\mathsf{single}})\right] - 1/2\right|.$$

▷ Note that $q = q(n)$ is a polynomial, and the advantage of $\mathcal{A}_{\mathsf{single}}$ has to be negligible. □.

- **Corollary**:

  If $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a fixed-length (say $\mathcal{M}_n = \{0,1\}^n$) CPA-secure ES,

  then $\mathcal{E}' = (\mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}')$ with

  - "$\mathcal{M}_n' = (\{0,1\}^n)^*$", and $\mathsf{Enc}_k'(m) = \mathsf{Enc}_k(m^{(1)})||\ldots||\mathsf{Enc}_k(m^{(s)})$

  is also CPA-secure.

▷ **Ex**: This construction does not preserve CCA-security, i.e.

  assume $\mathcal{E}$ is CCA-secure with $\mathcal{M}_n = \{0,1\}^n$, and show that $\mathcal{E}'$ is not CCA-secure anymore (it is still CPA-secure).

- How to achieve CPA-security?

▷ Try to make the Enc-oracle useless, then we only have to achieve comp. secrecy.

- What means useless?

▷ Information that Eve could have generated herself.

▷ Recall: Perfect secrecy of the OTP

The ciphertext is useless to Eve as it is simply a random message which she thus could have generated herself.

- Ideally, we'd like to use a new OTP for every encryption:

  Then any two encryptions would be completely unrelated (independent), random messages, and Eve's oracle access to Enc_k would be useless.

▷ Problem: Infeasible in the real world; Alice&Bob would need to exchange key material for both their own messages and those chosen by Eve.

- Fix: Use a variable-length PRG G to emulate an "variable-length OTP".

▷ Recall: Essentially, a vlPRG G stretches a finite key $k \in \{0,1\}^n$

  into a stream (infinite sequence) $G(k; 1^\omega) \in \{0,1\}^\omega$.

  - but in finite time we can only see a finite prefix of $G(k; 1^\omega)$

  - $G(k; 1^s)$ are the first s bits of $G(k; 1^\omega)$.

  - Definition of vlPRG requires that for any fixed polynomial $l(n)$

    $$G_l(x) := G(x, 1^{l(|x|)})$$

    is a PRG.

  ▷ E.g. let $l(n)$ be the run time of Eve's attack!

  - Hence, any part of $G_l(x)$ is indistinguishable from a truly random string to PPT-Eve.

▷ **Definition**: stateful counter mode (sCTR) for a vl-PRG G

For $s \leq s'$ and $G(x, 1^{s'}) = y_1 y_2 \ldots y_s \ldots y_{s'}$

let $G(x)[s, s'] = y_s \ldots y_{s'}$. (Recall the "prefix property" of G).

- $\mathcal{K}_n = \{0,1\}^n$, $\mathcal{M}_n = \{0,1\}^+$, $\mathcal{C}_n = \{0,1\}^+$

- Gen: On input 1^n, output $k \overset{u}{\in} \mathcal{K}_n$.

- Enc: Static variable ctr; initially $\mathrm{ctr} = 0$.

  On input $k \in \mathcal{K}_n$ and $m \in \mathcal{M}_n$,

  compute $c = \lfloor \mathrm{ctr} \rfloor || m \oplus G(k)[\mathrm{ctr}+1, \mathrm{ctr}+|m|]$.
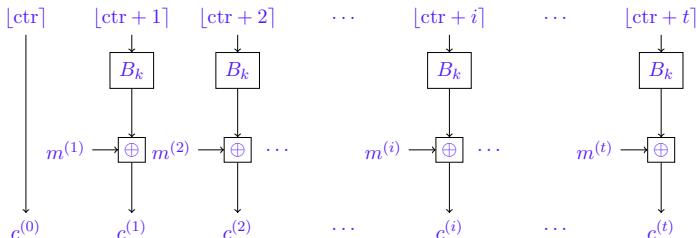
    - Where $\lfloor \cdot \rfloor$ encodes ctr as an n-bit string.

      ▷ So, encrypt no more than 2^n bits with a single k (throw an exception).

  Set $\mathrm{ctr} := \mathrm{ctr} + |m|$, then output c.

- Dec: On input $k \in \mathcal{K}_n$ and $c \in \mathcal{C}_n$, read the first n bits as the value of ctr, then decrypt the remaining bits.

- In practice, e.g. a block cipher B (block length l) is used for G:

  - E.g. $G(x, 1^{l \cdot s}) = B_k(\lfloor 1 \rceil) || B_k(\lfloor 2 \rceil) || \ldots || B_k(\lfloor s \rceil)$.

  - Then $\mathcal{M} = \left(\{0,1\}^l \right)^*$.

  - ▷ Remember in ctr the number of blocks encrypted so far.

- Advantage of using a block cipher as a PRG of variable stretch:

  - "random access" to the pseudorandom output stream: $B_k(\lfloor i \rceil)$ does not depend on $B_k(\lfloor i + 1 \rceil)$.

  - Allows to process message blocks in parallel.

- **Theorem**: G-sCTR is CPA-secure if G is a variable-length PRG.

▷ Proof sketch: Essentially the same as for the prOTP

From a PPT-adversary $\mathcal{A}$ for the game INDCPA vs. G-sCTR

- Let $T(n)$ be the running time of $\mathcal{A}$.

  ▷ $\mathcal{A}$ can only encrypt messages of length $T(n)$ in total.

we construct a PPT-distinguisher for G for fixed stretch $T(n)$.

- $\mathcal{D}$ simulates the interaction of $\mathcal{A}$ and G-sCTR in the game INDCPA.

  ▷ $\mathcal{D}$ gets as input $y = y_1 y_2 \ldots y_{T(n)}$.

  ▷ $\mathcal{D}$ mimics Enc by using y as replacement for $G(k)$:

  $\widetilde{\mathsf{Enc}}(m) := \lfloor \mathrm{ctr} \rfloor || m \oplus y[\mathrm{ctr} + 1, \mathrm{ctr} + |m|] ; \mathrm{ctr} + = |m| ;$

  with $\mathrm{ctr} = 0$ initially.

Alice&Bob	$\mathcal{D}$	sub: $\mathcal{A}$
$b \overset{u}{\in} \{0,1\}$		
$b = 0$: $y \overset{u}{\in} \{0,1\}^{T(n)}$		
$b = 1$: $x \overset{u}{\in} \{0,1\}^n$, $y = G(x, 1^{T(n)})$		
pass y to $\mathcal{D}$		
	run $\mathcal{A}^{\widetilde{\mathsf{Enc}}}(1^n)$	
		return $m_0, m_1 \in \{0,1\}^n$
	$b' \overset{u}{\in} \{0,1\}$	
	$c = \widetilde{\mathsf{Enc}}(m_{b'})$	
	run $\mathcal{A}^{\widetilde{\mathsf{Enc}}}(1^n, c)$	
		return r'
	return $r := (r' \overset{?}{=} b')$	

- If $b = 1$, then $\widetilde{\mathsf{Enc}}$ coincides with Enc of G-sCTR.

  - $\mathcal{D}$ wins iff $\mathcal{A}$ wins vs. G-sCTR in INDCPA.

- If $b = 0$, then $\widetilde{\mathsf{Enc}}$ uses for every message a fresh OTP.

  - $\mathcal{A}$ gets as ciphertexts only truly random strings.

  - $\mathcal{D}$ wins iff $\mathcal{A}$ loses vs. the "OTP" in INDED:

- Hence

$$\Pr\left[\mathsf{Win}_{n,G}^{\mathrm{IndPRG}}(\mathcal{D}) \mid b = 1\right] = \Pr\left[\mathsf{Win}_{n,G\text{-}\mathsf{sCTR}}^{\mathrm{IndCPA}}(\mathcal{A})\right]$$

- and

$$\Pr\left[\mathsf{Win}_{n,G}^{\mathrm{IndPRG}}(\mathcal{D}) \mid b = 0\right] = \frac{1}{2}$$

- So in total:

$$\Pr\left[\mathsf{Win}_{n,G}^{\mathrm{IndPRG}}(\mathcal{D})\right] = \frac{1}{2} \cdot \Pr\left[\mathsf{Win}_{n,G\text{-}\mathsf{sCTR}}^{\mathrm{IndCPA}}(\mathcal{A})\right] + \frac{1}{4}$$

- As the advantage of $\mathcal{D}$ has to be negligible, we obtain that

$$\left|\Pr\left[\mathsf{Win}_{n,G\text{-}\mathsf{sCTR}}^{\mathrm{IndCPA}}(\mathcal{A})\right] - \frac{1}{2}\right| = 2\left|\Pr\left[\mathsf{Win}_{n,G}^{\mathrm{IndPRG}}(\mathcal{D})\right] - \frac{1}{2}\right|$$

the advantage of $\mathcal{A}$ has to be negligible, too.

- **Lemma**: G-sCTR is not CCA-secure:

▷ Proof: Consider the following $\mathcal{A}_{\text{IndCCA}}$

    ❶ Output $m_0 = 0^n$ and $m_1 = 1^n$.

    ❷ Receive $c = \lfloor \text{ctr} \rfloor || c'$.

    ❸ Flip the last bit of c and, thus, of c': $\tilde{c} = c \oplus 0^n 0^{n-1} 1$.

    ❹ As $\tilde{c} \neq c$, query the decryption oracle for an decryption.

    ❺ Return 0 iff the oracle returns $0^{n-1} 1$.

▷ The crucial step is that $\mathcal{A}_{\text{IndCCA}}$ can easily forge a new ciphertext.

▷ Idea: use a MAC so that Alice&Bob resp. the decryption oracle can authenticate the origin of a ciphertext.

▷ The MAC will allow the decryption oracle to reject ciphertexts forged by Eve, thus, forcing her back to the CPA-setting.

▷ "CPA-secure ES + secure MAC = CCA-secure ES"

[1] N. Ferguson, B. Schneier, and T. Kohno. *Cryptography Engineering –*
     Chapter 9. URL: https://www.schneier.com/fortuna.pdf.
[2] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*.