Lecture 2: stream ciphers

<u>Recap</u>: A cipher defined over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ is a pair of "eff" algs. (E, D) s.t.

$$\forall m \in \mathcal{M}, \forall k \in \mathcal{K} : \quad D(k, E(k,m)) = m$$

weak cipher: subs. cipher (CT only attack)

good cipher: OTP $\quad \mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$

(super fast) $\quad E(k,m) = k \oplus m, \quad D(k,c) = k \oplus c$

what is a secure cipher?

Shannon: cipher (E, D) has <u>perfect secrecy</u> if <u>single</u> CT reveals no "info" about PT (other than its length) when $k \xleftarrow{R} \mathcal{K}$

<u>Thm</u>: OTP has perfect secrecy

$\Longrightarrow$ no CT only attack (if key is only used once)

The <u>two time pad</u> is insecure !!

$$\begin{cases} c_1 \leftarrow m_1 \oplus k \\ c_2 \leftarrow m_2 \oplus k \end{cases} \Rightarrow c_1 \oplus c_2 = m_1 \oplus m_2$$

$\Rightarrow$ can recover m_1, m_2 (HW #1)

<u>Thm</u>: (bad news) perfect secrecy $\Rightarrow |\mathcal{K}| \geq |\mathcal{M}|$

so: key-len $\geq$ msg-len

Proof: exercise. Fix $c \in \mathcal{C}$. show that

$\forall\, m \in \mathcal{M}$ there is a distinct $k \in \mathcal{K}$ s.t. $E(k, m) = c$.

Can we make the OTP useful?

stream ciphers

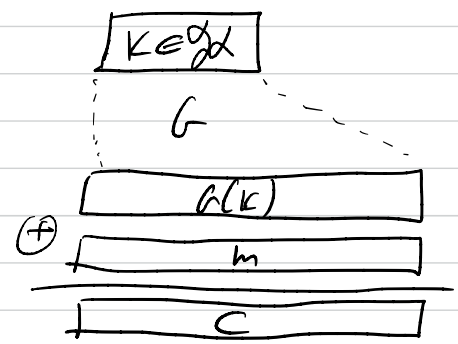
idea: replace "random" key by a "pseudorandom" key

Def: PRG is a function $G: \underbrace{\{0,1\}^s}_{\text{seed}} \longrightarrow \underbrace{\{0,1\}^n}_{\text{output}}$ $\quad n \gg s$

where G is "eff" computable by a det. alg.

Def: The stream cipher (E, D) derived from G is defined over $\mathcal{C} = \mathcal{M} = \{0,1\}^n$, $\mathcal{K} = \{0,1\}^s$

$$\begin{cases} C = E(k,m) := m \oplus G(k) \\ D(k,C) := C \oplus G(k) \end{cases}$$



$|\mathcal{K}| < |\mathcal{M}| \Rightarrow$ no perfect secrecy

- Need a different security definition
- Security depends on PRG

What is a secure PRG?

Let $X = \{0,1\}^s$

Consider $G : X \to X^n$ where $G(k) = \underbrace{(k, k, \ldots, k)}_{n \text{ times}}$

When $k \xleftarrow{R} X$, does $G(k)$ "look" random?

No! We can distinguish the output from random.

Define $A(y \in X^n)$: $\boxed{\begin{array}{l} \text{if } y[0] = y[1] \text{ output } 0 \text{ (not random)} \\ \text{else output } 1 \text{ (random)} \end{array}}$

Then: when $k \xleftarrow{R} X$, $r \leftarrow G(k)$: $\Pr[A(r) = 1] = 0$

when $r \xleftarrow{R} X^n$: $\Pr[A(r) = 1] = 1 - \frac{1}{2^s}$

huge gap!

We say that A distinguishes random from pseudorandom.

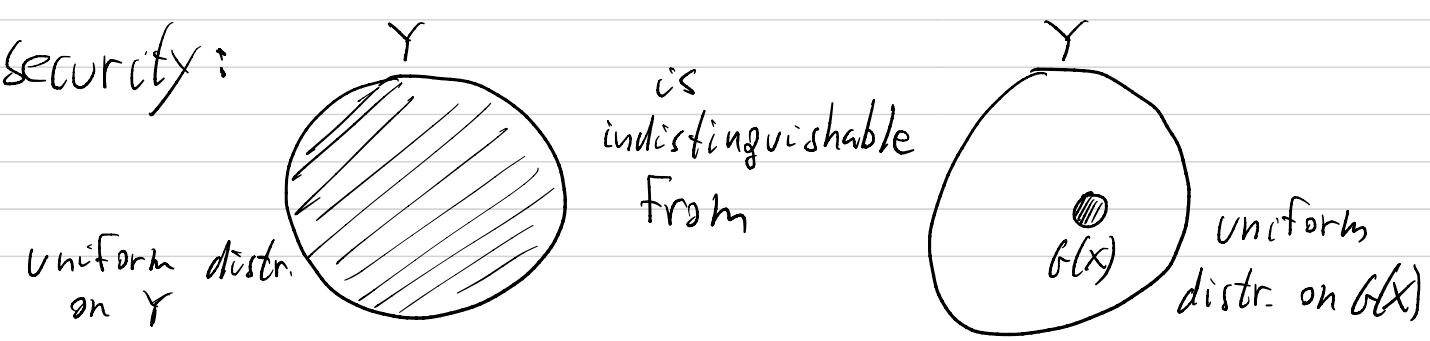
Define the advantage of A in attacking G as:

$$\mathrm{Adv}[A, G] := \left| \Pr\left[\begin{array}{l} k \xleftarrow{R} X \\ r \leftarrow G(k) \end{array} : A(r) = 1 \right] - \Pr\left[r \xleftarrow{R} Y : A(r) = 1 \right] \right| \in [0,1]$$

Def: $G : X \to Y$ is a secure PRG if for every "eff" A

$\mathrm{Adv}[A, G] \le \text{negl.}$ $[= 2^{-80}]$

PRG security:



$\Rightarrow$ every "eff" $\mathcal{A}$ behaves the same on $\{r \xleftarrow{R} Y\}$ (uniform dist. on Y)
as on $\{G(k \xleftarrow{R} X)\}$

$\Rightarrow$ if $k \xleftarrow{R} X$, we can treat $G(k)$ <u>as if</u> it were uniform on Y.

... exactly what we need for a stream cipher.

<u>Thm:</u> PRG is secure $\Rightarrow$ derived stream cipher is "secure" against one-time CT only attack
(semantic security - will define next week)

<u>More generally:</u> Fix a finite set Ω.
Let P_1, P_2 be two distributions over Ω.
P_1 & P_2 are <u>indist.</u> if for all "eff" $\mathcal{A}$:

$$\mathrm{Adv}[\mathcal{A}, (P_1, P_2)] := \left| \Pr[x \xleftarrow{R} P_1 : \mathcal{A}(x) = 1] - \Pr[x \xleftarrow{R} P_2 : \mathcal{A}(x) = 1] \right| \leq \text{negl.}$$
$[= 2^{-80}]$

$\Rightarrow$ every "eff" alg. behaves on P_1 the same as on P_2.

An important property of a PRG: unpredictable

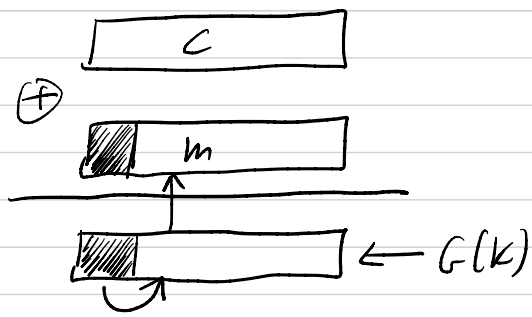
We say that $G: \{0,1\}^s \to \{0,1\}^n$ is <u>predictable</u>
if $\exists$ "eff" $\mathcal{A}$ and $i \in \{1,\dots,n\}$ s.t.

(*) $\quad \forall k \in \{0,1\}^s : \quad r \leftarrow G(k), \quad \mathcal{A}(r[0,\dots,i-1]) = r[i]$

ex: $G(k) = (k, \dots, k)$ is predictable

G is predictable $\Longrightarrow$ derived stream cipher is insecure

why?



https://

<u>Thm:</u> G is a secure PRG $\Longrightarrow$ G is not predictable

Proof: contra positive.

G predictable $\Longrightarrow$ $\exists$ "eff" $\mathcal{A}$ & $i \in \{1,\dots,n\}$ as (*).

We use $\mathcal{A}$ to construct a distinguisher $\mathcal{B}$ for G:

Alg $\mathcal{B}$: input $r \in \{0,1\}^n$
(1) run $\mathcal{A}(r[0,\dots,i-1]) \longrightarrow b \in \{0,1\}$ (prediction)
(2) if $b == r[i]$ output 0; else output 1

<u>claim:</u> $Adv[\mathcal{B}, G] = |0 - \frac{1}{2}| = \frac{1}{2}$ (not negligible)

$\Longrightarrow$ G is not a secure PRG. ∎

An example secure PRG

First bad PRGs: (do not use for crypto)

(1) glibc rand()
(2) Java Math.random()

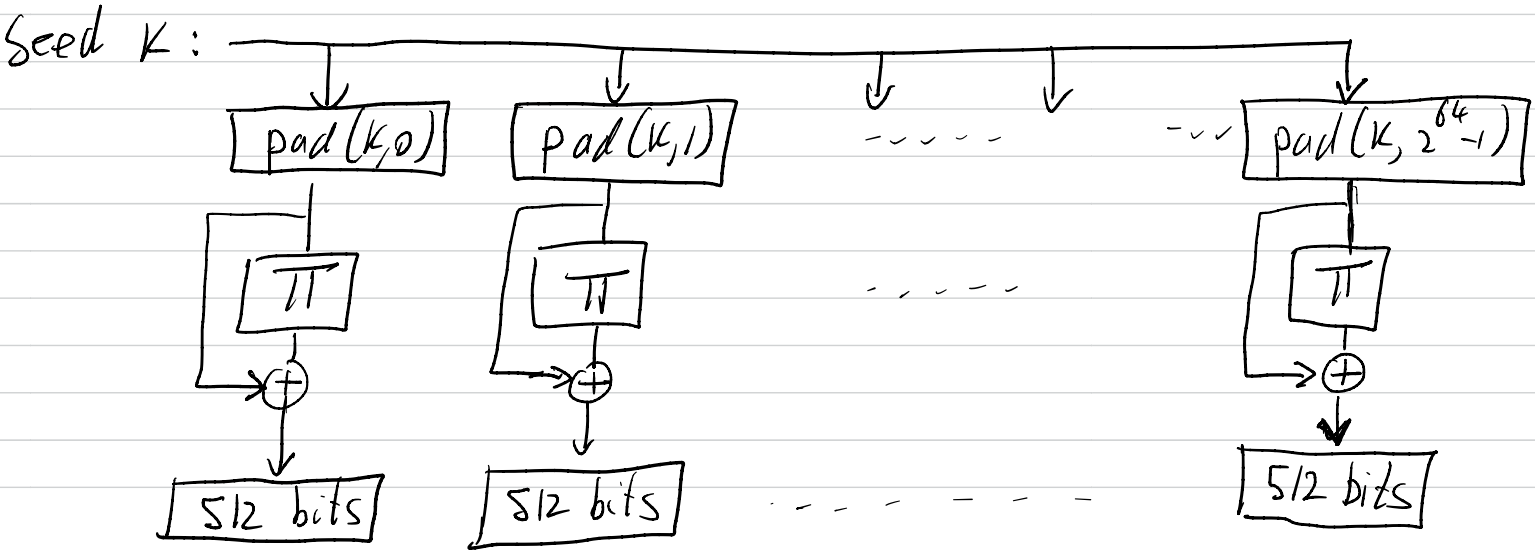
] predictable generators

Chacha20: a secure PRG (as far as we know)
fast on 64-bit processors.
widely used on Android.

$\pi : \{0,1\}^{512} \to \{0,1\}^{512}$ a <u>fixed</u> one-to-one function

$k \in X = \{0,1\}^{256}$, $j \in \{0,1,\ldots,2^{64}-1\}$

$$\text{pad}(\underbrace{k, j}_{256+64=330 \text{ bits}}) \longrightarrow \{0,1\}^{512}$$

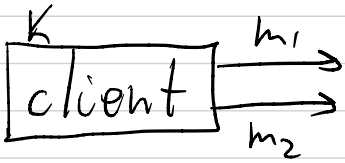


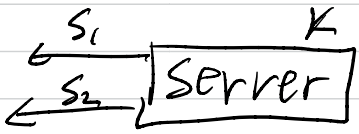
Attacks on OTP & stream ciphers

(1) <u>Two time pad attack:</u>

never use a stream cipher key more than once.

example mistake: MS-PPTP





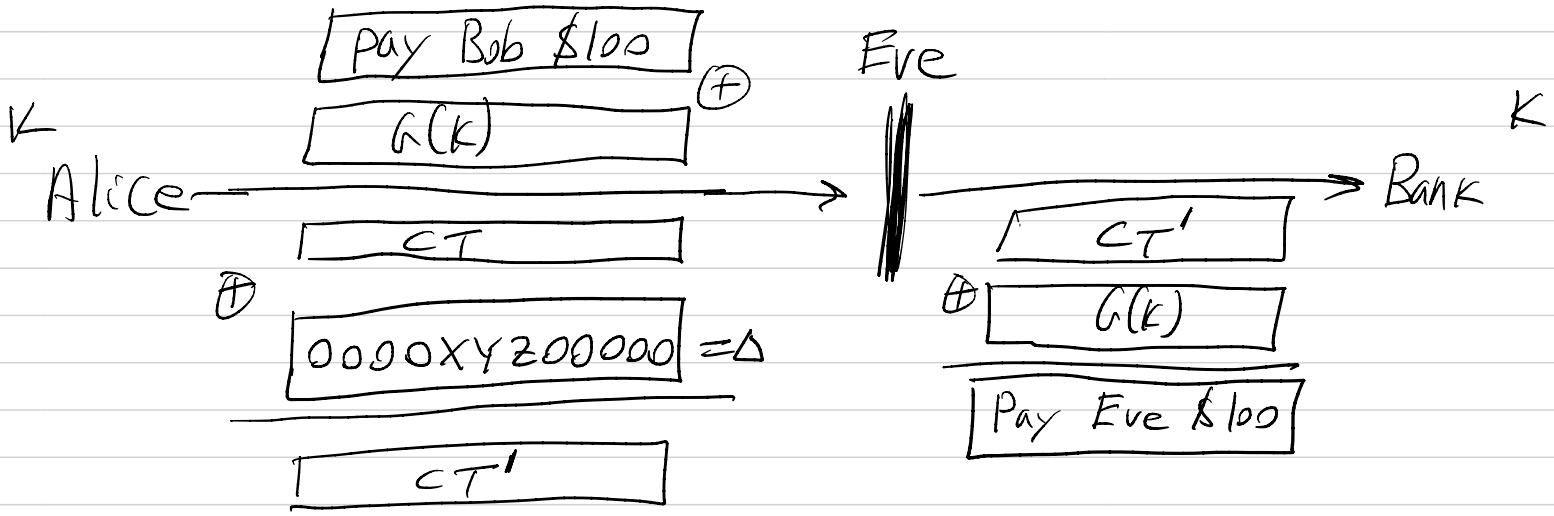
$[m_1, m_2, \ldots] \oplus G(k)$ $\qquad$ $[s_1, s_2] \oplus G(k)$

(also in 802.11b WEP)

---

(2) <u>No integrity:</u>

$$\Big[\underbrace{(m \oplus G(k))}_{CT} \oplus \Delta\Big] \underbrace{\oplus G(k)}_{\text{decrypt}} = m \oplus \Delta$$



Bob: 42 6F 62 $\qquad$ Eve: 45 76 65 $\qquad$ $XYZ = \text{Bob} \oplus \text{Eve} = \boxed{07\ 19\ 07}$