# CSCI 1510

- Merkle-Damgård Transform
- Hash-and-MAC
- Applications of Hash Functions
- Constructions of Block Cipher

# Collision-Resistant Hash Function (CRHF)

- **Syntax:**

  A hash function is defined by a pair of PPT algorithms $(Gen, H)$:

  - $Gen(1^n)$: output $s$
  - $H^s(x)$: $x \in \{0,1\}^*$, output $h \in \{0,1\}^{\ell(n)}$

- **Security**

  A hash function $(Gen, H)$ is **collision-resistant** if

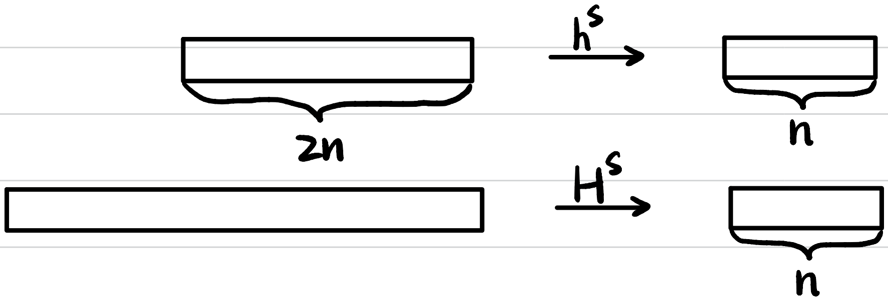
  $\forall$PPT $\mathcal{A}$, $\exists$negligible function $\varepsilon(\cdot)$ s.t. $\Pr[x \neq x' \wedge H^s(x) = H^s(x')] \leq \varepsilon(n)$.

| $C(1^n)$ | | $\mathcal{A}(1^n)$ |
|---|---|---|
| $s \leftarrow Gen(1^n)$ | | |
| | $\xrightarrow{s}$ | |
| | | output $(x, x')$ |

# Domain Extension: Merkle–Damgård Transform

Given a CRHF (Gen, h) from $\{0,1\}^{2n}$ to $\{0,1\}^n$,

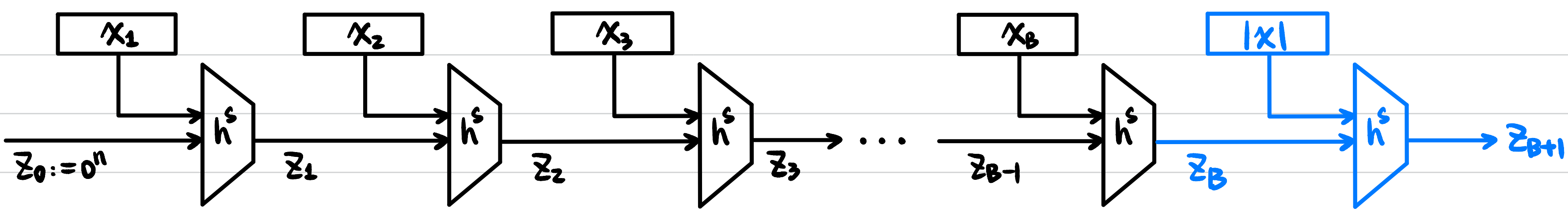
Construct a CRHF (Gen, H) from $\{0,1\}^*$ to $\{0,1\}^n$.




- $\mathrm{Gen}(1^n)$: remains unchanged.
- $H^s(x)$: $x \in \{0,1\}^*$

① Pad $x$ with $100\cdots0$ to a multiple of $n$ $\rightarrow \tilde{x}$

② Parse $\tilde{x} = x_1 \| x_2 \| \cdots \| x_B$, $\quad x_i \in \{0,1\}^n \quad \forall i \in [B]$



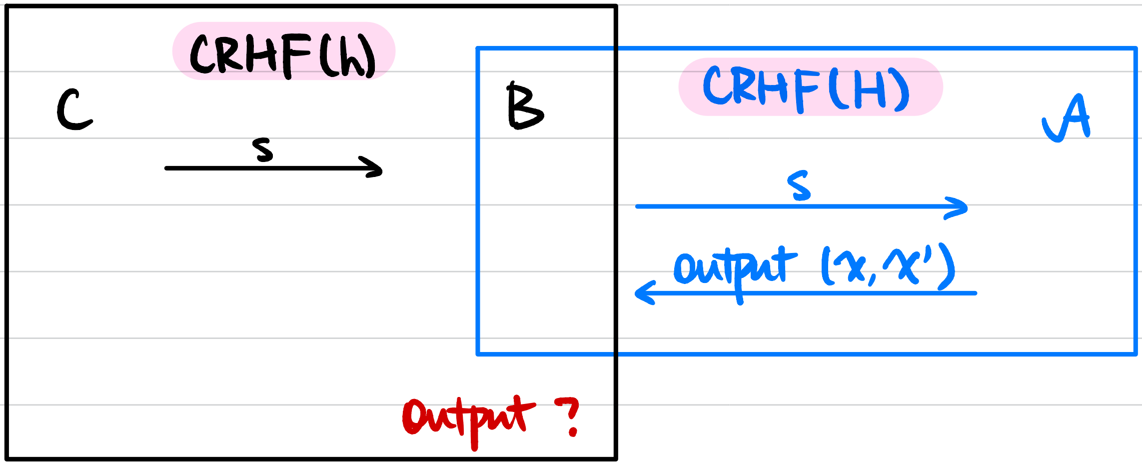

bit representation of $|x|$

$z_0 := 0^n \qquad z_i := h^s(z_{i-1} \| x_i) \quad \forall i \in [B]. \qquad z_{B+1} := h^s(z_B \| \langle \ell \rangle) \qquad H^s(x) := z_{B+1}$

<u>Thm</u> If (Gen, h) is CRHF, then so is (Gen, H).

**Thm** If $(\mathrm{Gen}, h)$ is CRHF, then so is $(\mathrm{Gen}, H)$.

**Proof** Assume not, then $\exists$ PPT $\mathcal{A}$ that breaks the collision resistence of $(\mathrm{Gen}, H)$.
We construct a PPT $B$ to break the collision resistence of $(\mathrm{Gen}, h)$.




Collision:
$x \neq x' \wedge H^s(x) = H^s(x')$

Case 1: $|x| \neq |x'|$. $h^s(z_B \| \langle l \rangle) = h^s(z'_{B'} \| \langle l' \rangle)$

$m := z_B \| \langle l \rangle \qquad m' := z'_{B'} \| \langle l' \rangle$

Case 2: $|x| = |x'|$. $B = B'$

If $z_B \neq z'_B$: $\quad m := z_B \| \langle l \rangle \qquad m' := z'_B \| \langle l \rangle$.

If $z_B = z'_B$: $\quad \exists i \in [B]$ s.t. $x_i \neq x'_i \wedge z_i = z'_i$

$m := z_{i-1} \| x_i \qquad m' := z'_{i-1} \| x'_i$

$B$ outputs $(m, m')$ as a collision for $h^s$.

# Hash-and-MAC

Secure MAC for fixed-length messages
+
CRHF for arbitrary-length inputs
$\Rightarrow$ Secure MAC for arbitrary-length messages

Let $\Pi^M = (\mathrm{Gen}^M, \mathrm{Mac}^M, \mathrm{Vrfy}^M)$ be a secure MAC for messages of length $n$.
Let $\Pi^H = (\mathrm{Gen}^H, H)$ be a CRHF for arbitrary-length inputs with output length $n$.

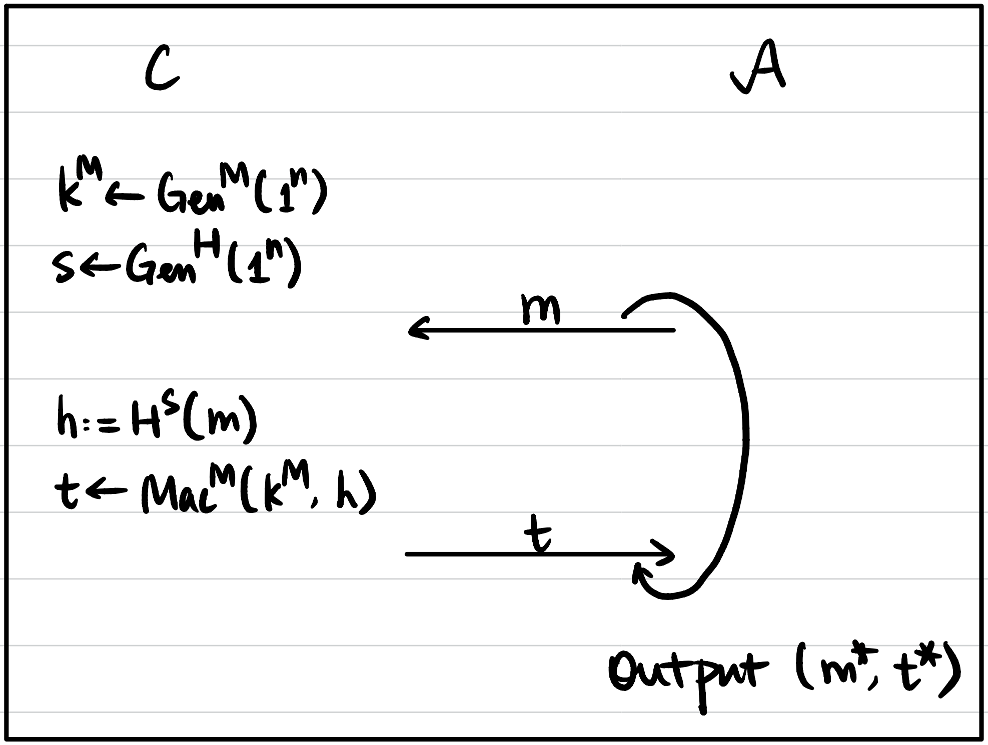
Construct $\Pi = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$:

- $\mathrm{Gen}(1^n)$: $k^M \leftarrow \mathrm{Gen}^M(1^n)$, $s \leftarrow \mathrm{Gen}^H(1^n)$. Output $k = (k^M, s)$
- $\mathrm{Mac}(k, m)$: $m \in \{0,1\}^*$, parse $k = (k^M, s)$
  $h := H^s(m)$, $t \leftarrow \mathrm{Mac}^M(k^M, h)$. Output $t$.
- $\mathrm{Vrfy}(k, (m, t))$: Parse $k = (k^M, s)$
  $h := H^s(m)$, $b := \mathrm{Vrfy}^M(k^M, (h, t))$. Output $b$.

$$\boxed{\qquad\qquad m \qquad\qquad} \xrightarrow{H^s} \boxed{\quad h \quad} \xrightarrow{\mathrm{Mac}^M} \boxed{\quad t \quad}$$

**Thm** If $\Pi^M$ is a secure MAC and $\Pi^H$ is CRHF, then $\Pi$ is a secure MAC.

**Thm** If $\Pi^M$ is a secure MAC and $\Pi^H$ is CRHF, then $\Pi$ is a secure MAC.




$Q := \{m \mid m \text{ queried by } \mathcal{A}\}$

**Step 1:** $\forall$ PPT $\mathcal{A}$, $\Pr[\exists m \in Q \text{ s.t. } m \neq m^* \wedge H^s(m) = H^s(m^*)] \leq negl(n)$.
(follows from collision-resistance of H)

**Step 2:** Assume $H^s(m) \neq H^s(m^*)$ $\forall m \in Q$, then unforgeability follows from MAC security.

# Applications of Hash Functions

- **Deduplication**

$$H(\boxed{D_1}) \rightarrow h_1$$

unique identifier

$$H(\boxed{D_2}) \rightarrow h_2$$

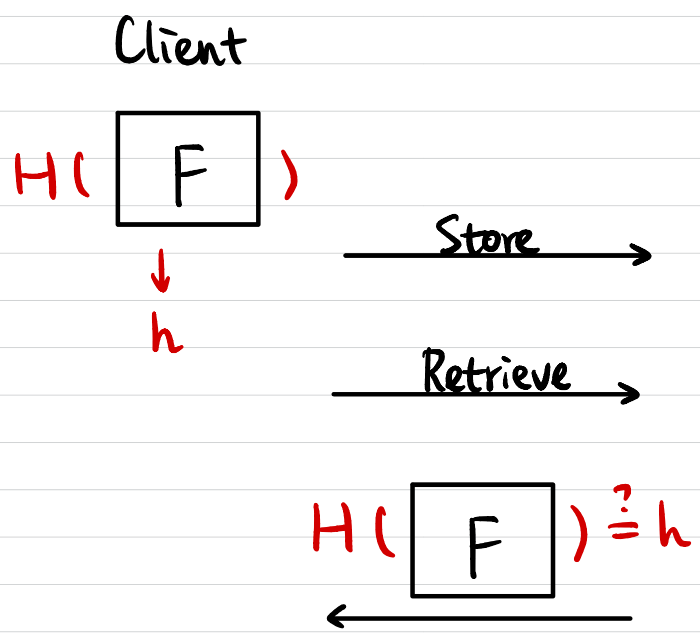
If $h_1 \neq h_2 \Rightarrow D_1 \neq D_2$

If $h_1 = h_2 \Rightarrow D_1 = D_2$ Why?

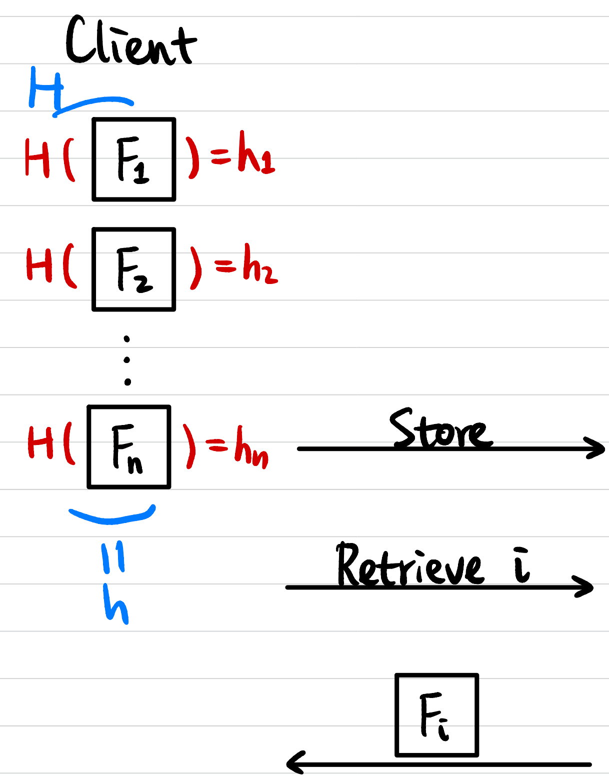
**Virus Scan** $H(\boxed{F}) \stackrel{?}{=} H(\boxed{F^*})$

**Video Deduplication** $H(\boxed{V_1}) \stackrel{?}{=} H(\boxed{V_2})$

# Applications of Hash Functions




Is the file changed?




Is the file changed?

**Goal**:

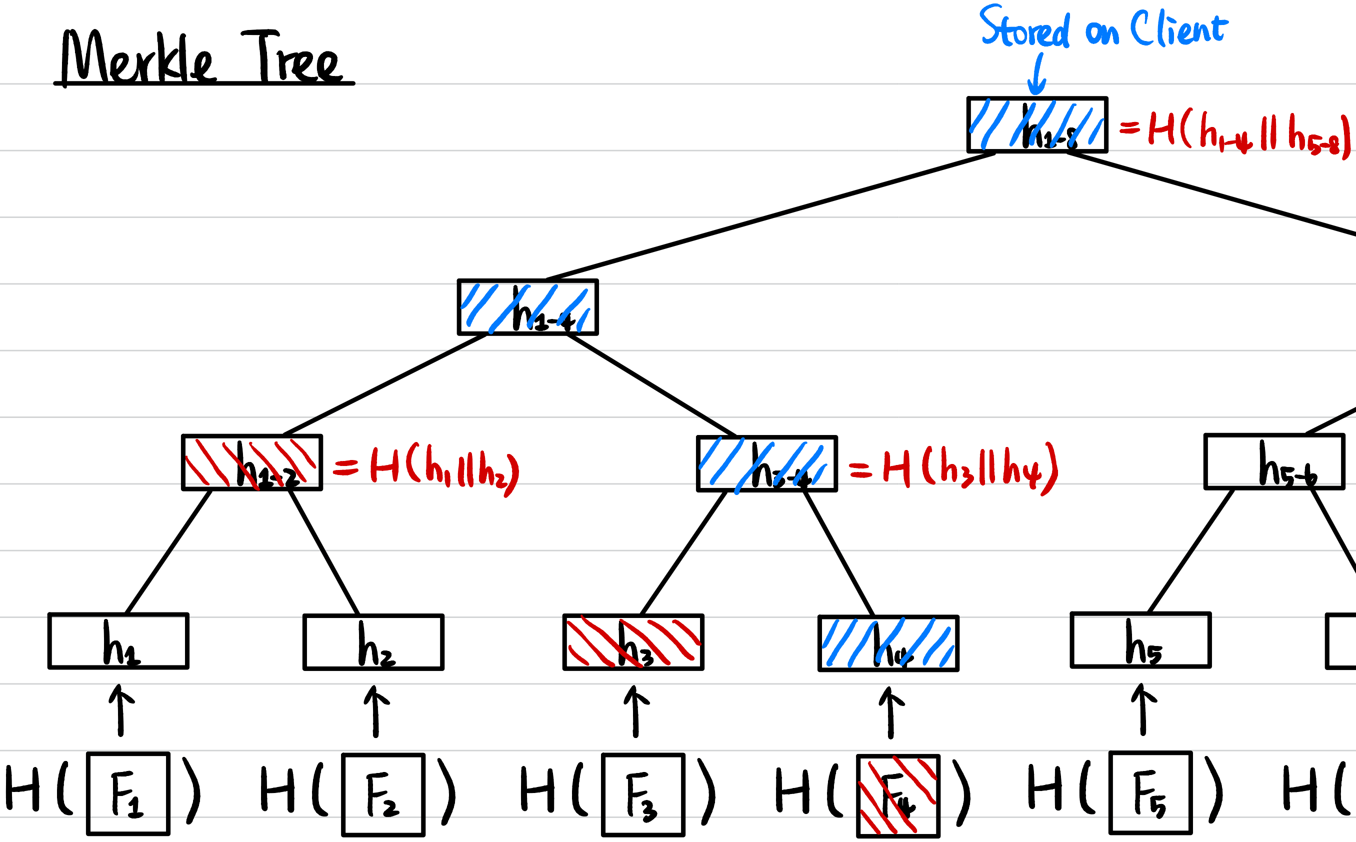
$O(1)$

① Client's storage doesn't grow with n.

② Verification doesn't grow with n.

$O(\log n)$

# Merkle Tree




$$H^s : \{0,1\}^* \to \{0,1\}^n$$

$$MT_t^s(F_1 \| \cdots \| F_t) \to \{0,1\}^n$$

How does verification work?

**Thm** If $(Gen, H)$ is a CRHF, then $(Gen, MT_t)$ is a CRHF for any fixed $t = 2^k$.

$$
\begin{tikzcd}
 & & \text{CPA-Enc} \arrow[dr] & & \text{CCA (Secrecy)} \\
 & \text{PRF} \arrow[ur] \arrow[dr] & & \text{Authenticated Encryption} \arrow[ur, dash] \arrow[dr, dash] & \\
\text{How to construct?} \arrow[ur, bend left] \arrow[dr, bend right] & & \text{MAC} \arrow[ur] & & \text{Unforgeable (Integrity)} \\
 & \text{Hash Functions} & & &
\end{tikzcd}
$$

# Pseudorandom Function (PRF)

**Def** Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a deterministic, poly-time, keyed function. $F$ is a pseudorandom function (PRF) if $\forall$ PPT $\mathcal{A}$, $\exists$ negligible function $\varepsilon(\cdot)$ s.t.

$$\left| \Pr_{k \leftarrow U_n}\left[\mathcal{A}^{F_k(\cdot)}(1^n) = 1\right] - \Pr_{f \xleftarrow{\$} Func_n}\left[\mathcal{A}^{f(\cdot)}(1^n) = 1\right] \right| \leq \varepsilon(n)$$

$C(1^n)$ | $\mathcal{A}(1^n)$

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

If $b=0$, then $k \xleftarrow{\$} \{0,1\}^n$

If $b=1$, then $f \xleftarrow{\$} Func_n$

$\xleftarrow{\quad x \quad}$

If $b=0$, then $y := F_k(x)$

If $b=1$, then $y := f(x)$

$\xrightarrow{\quad y \quad}$

output $b'$

$$\Pr[b=b'] \leq \frac{1}{2} + \varepsilon(n).$$

# Pseudorandom Permutation (PRP)

**Def** Let $F: \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a deterministic, poly-time, keyed function. $F$ is a pseudorandom permutation (PRP) if $F_k(\cdot)$ is bijective for all $k$, $\forall$ PPT $\mathcal{A}$, $\exists$ negligible function $\varepsilon(\cdot)$ s.t.

$$\left| \Pr_{k \leftarrow U_n}\left[\mathcal{A}^{F_k(\cdot)}(1^n) = 1\right] - \Pr_{f \xleftarrow{\$} \mathrm{Perm}_n}\left[\mathcal{A}^{f(\cdot)}(1^n) = 1\right] \right| \le \varepsilon(n)$$

$C(1^n)$ $\qquad\qquad$ $\mathcal{A}(1^n)$

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

If $b=0$, then $k \xleftarrow{\$} \{0,1\}^n$

If $b=1$, then $f \xleftarrow{\$} \mathrm{Perm}_n$ $\quad$ $\mathrm{Perm}_n = \{F \mid F: \{0,1\}^n \leftrightarrow \{0,1\}^n\}$

$\xleftarrow{\quad x \quad}$

If $b=0$, then $y := F_k(x)$

If $b=1$, then $y := f(x)$

$\xrightarrow{\quad y \quad}$

output $b'$

$$\Pr[b = b'] \le \frac{1}{2} + \varepsilon(n).$$

# Block Cipher

$F: \{0,1\}^n \times \{0,1\}^l \rightarrow \{0,1\}^l$

$n$: key length

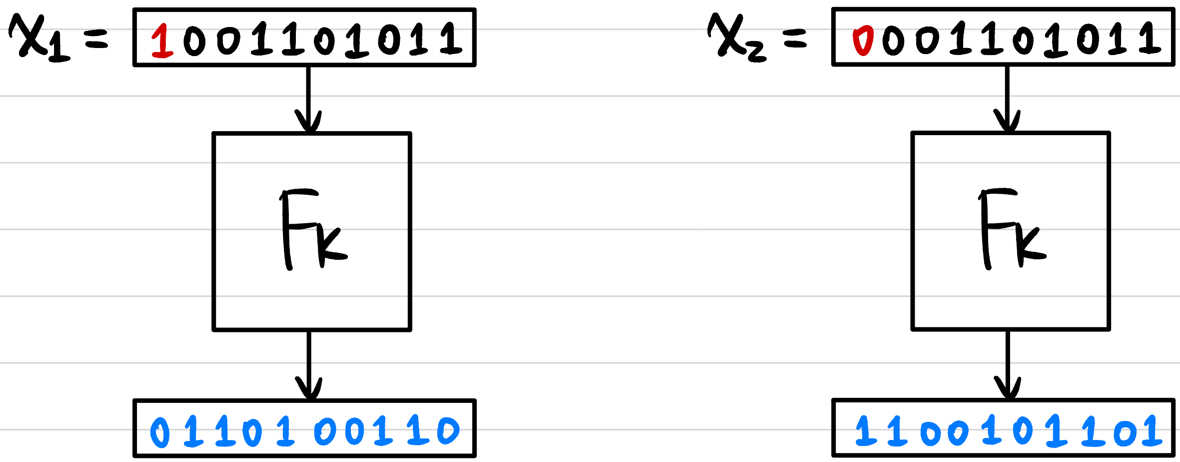
$l$: block length

$F_k(\cdot)$: permutation / bijective $\{0,1\}^l \rightarrow \{0,1\}^l$

$F_k^{-1}(\cdot)$: efficiently computable given $k$.

Assumed to be a pseudorandom permutation (PRP).

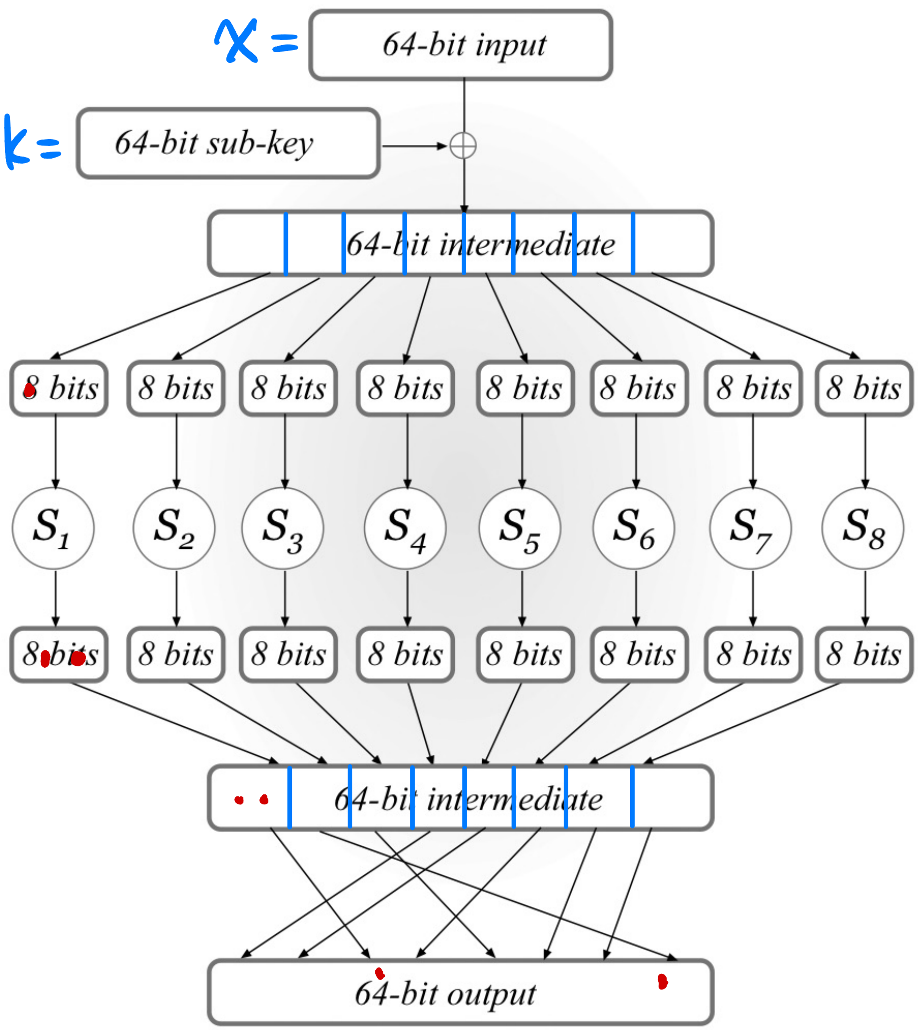
# Substitution-Permutation Network (SPN)




**Design Principle:** "Avalanche Effect"

A one-bit change in the input should "affect" every bit of the output.

# Substitution-Permutation Network (SPN)




A single round of SPN

"Confusion-Diffusion Paradigm"

**Step 1:** Key Mixing

$x := x \oplus k$

**Step 2:** Substitution (Confusion Step)

$S_i : \{0,1\}^8 \to \{0,1\}^8$ (S-box)

Public permutation / one-to-one map

1-bit change of input

→ at least 2-bit change of output

**Step 3:** Permutation (Diffusion Step)

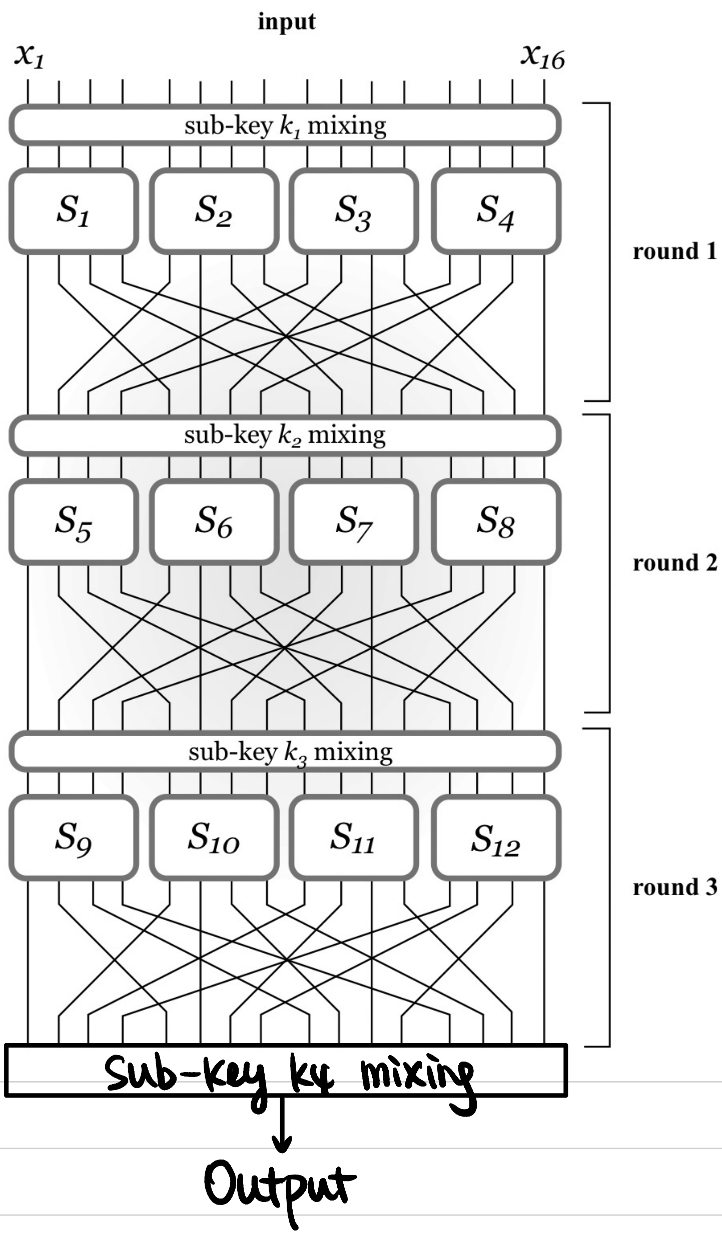
$P : [64] \to [64]$

Public mixing permutation

↓

affect input to multiple S-boxes next round

# Substitution-Permutation Network (SPN)




**3-round SPN:**

3-round
- key mixing
- substitution
- permutation

1 final-round key mixing

**Key Schedule:**

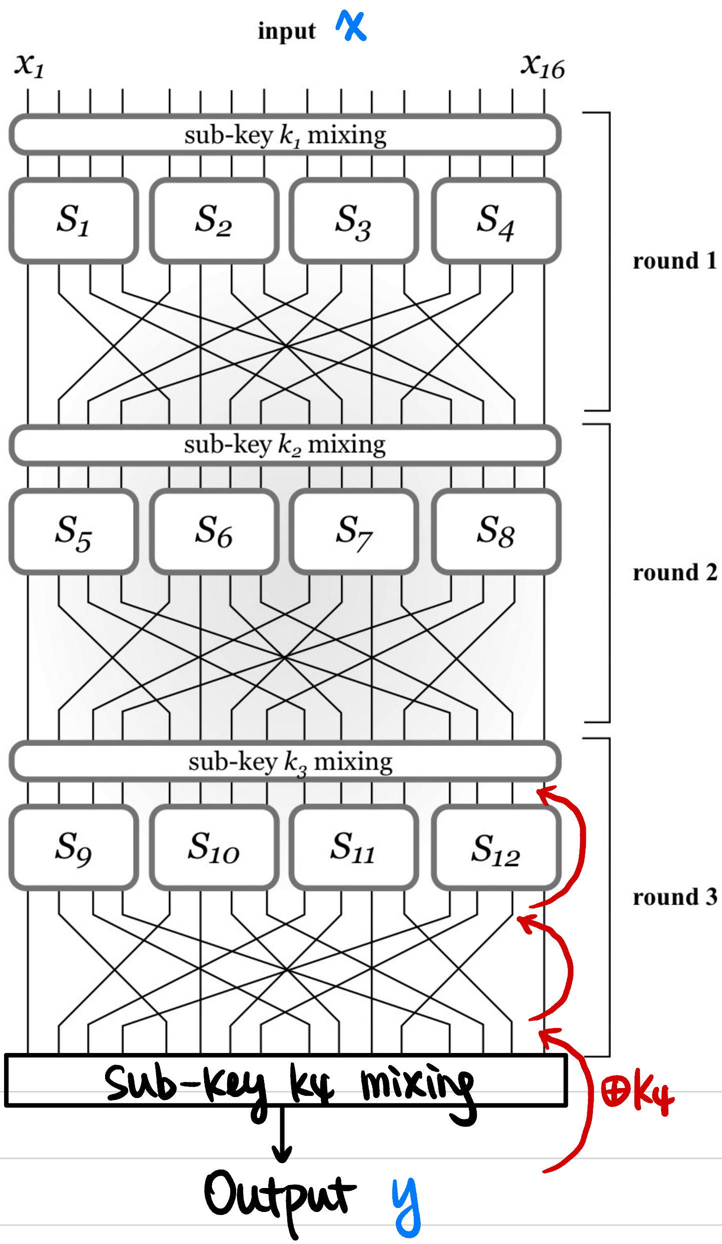
How we derive sub-keys from master key.

**Example:**

$$\underbrace{\boxed{\;k_1\;}\boxed{\;k_2\;}\boxed{\;k_3\;}\boxed{\;k_4\;}}_{\text{master key}}$$

$\uparrow$ $F_k(\cdot)$

# Substitution-Permutation Network (SPN)



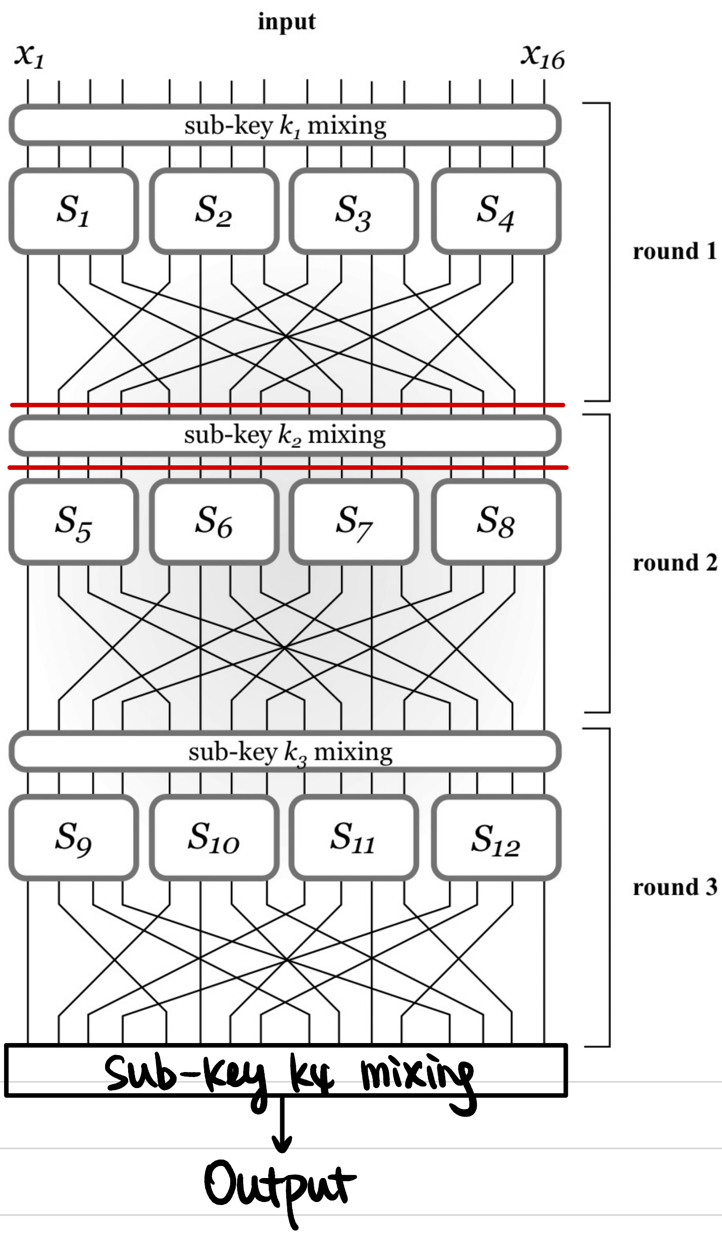

An SPN is invertible given the master key.

$\Downarrow$

permutation

How to compute $F_k^{-1}(y)$ ?

# Attacks on Reduced-Round SPN




1-round SPN without final key mixing?

$C \xleftarrow{x} A$

$\xrightarrow{y}$

$\Rightarrow k_1$

1-round SPN with final key mixing?

$C \xleftarrow{x} A$

$\xrightarrow{y}$

brute force search on $k_1 \Rightarrow k_2$

$O(2^{16})$

Why do we need a final key mixing step?

Can we do r-round key mixing, then r-round substitution, then r-round permutation?