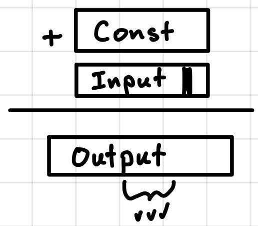
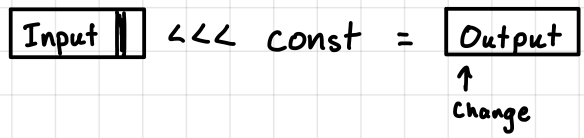
A low-level cryptographic function usually has:

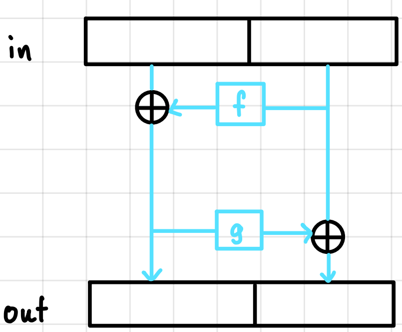
- Multiple simple steps over multiple mathematical structure
- provide "confusion": complex input-output relation
- provide "diffussion": changes in one part of the input affects parts of the output further away



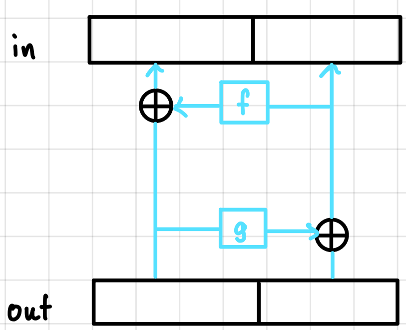


- multiple iterations (or "rounds") of the above

Feistel Structure

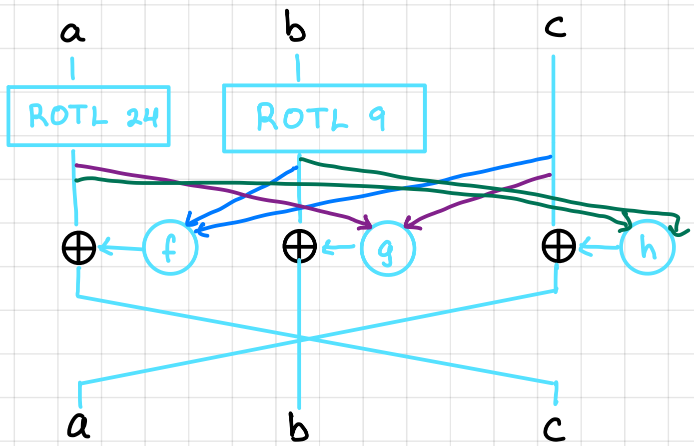


f and g don't have to be invertible

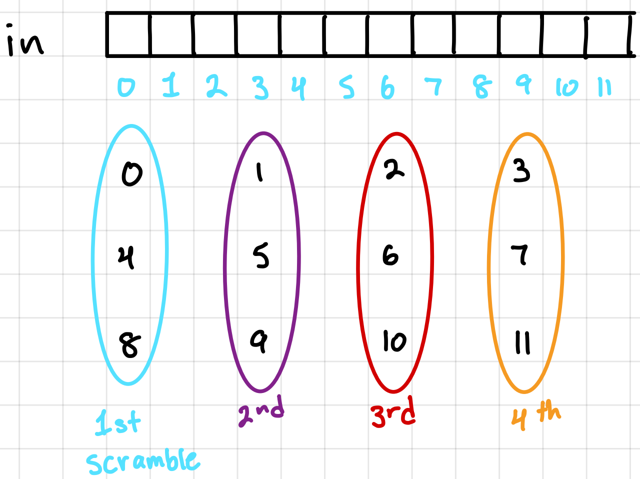


perm384 : $\{0,1\}^{384} \to \{0,1\}^{384}$ set of all 384 bit strings.

scramble $(\underbrace{a, b, c}_{\text{32 bit each}})$



perm 384



12 × uint32_t

Total of 96 scrambles ≈ 1500 asm instructions

End result:

- perm384 behaves like a random permutation (in short c code)
- Kerchhoff's law: adversary knows all algorithms but not secret keys.
- no keys here: so per384 should be considered public knowledge

<u>Distinguishing Games:</u> If an adversary can't tell the difference between A and B, then they can be used interchangebly.

Raw perm384 vs fresh random permutation

$\boxed{\text{W1: Let } f = \text{perm384}}$ $\boxed{\text{W2: Let } f = \text{fresh random permutation}}$

Dist

$\boxed{\langle i \rangle = \text{binary representation of}}$

$\boxed{\begin{array}{l} \text{Distinguisher}(f): \\ \quad \text{if } f(\langle 0 \rangle) == \text{perm384}(\langle 0 \rangle) \\ \quad\quad \text{output "perm 384"} \\ \quad \text{else} \\ \quad\quad \text{output "random perm"} \end{array}}$

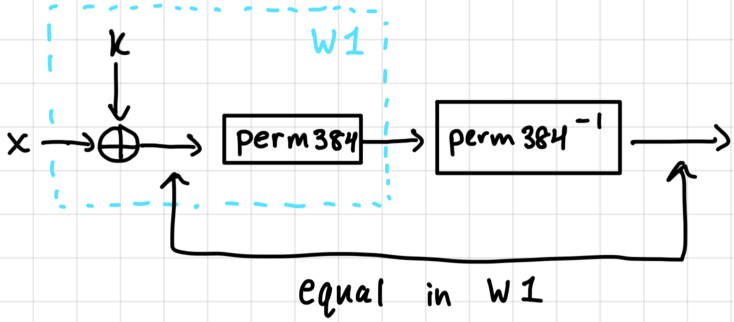
$$\begin{aligned}
\text{Advantage} &= \Pr[\text{right}] - \Pr[\text{wrong}] \\
&= \Pr[\text{output "perm384"} \mid f \text{ is perm384}] \\
&\quad - \Pr[\text{output "perm384"} \mid f \text{ is random perm}] \\
&= 1 - \frac{1}{2^{384}} \\
&\approx 1
\end{aligned}$$

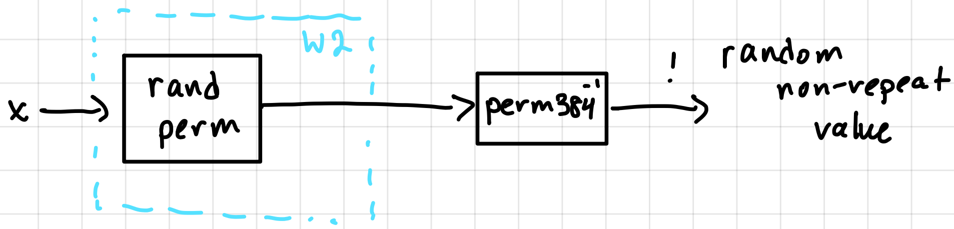
Scale of 1: Perfect
0: Awful

Try 2

W1: Let K be random 384 bit string

$$f(x) = \text{perm394}(x \oplus K)$$





Distinguish(f):

$Y_0 = f(\langle 0 \rangle)$

$Y_1 = f(\langle 1 \rangle)$

$X_0 = \text{perm384}^{-1}(Y_0)$

$X_1 = \text{perm384}^{-1}(Y_1)$

If$(x_1 \oplus x_0 == \langle 1 \rangle)$

output perm384

else

output random_perm

$$\langle 0 \rangle \to \overset{K \downarrow}{\oplus} \to \text{perm384}$$

$$X_0 = \langle 0 \rangle \oplus K$$

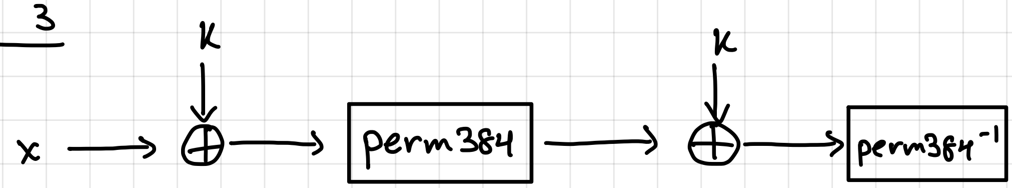
$$\langle 1 \rangle \to \overset{K \downarrow}{\oplus} \to \text{perm384}$$

$$x_1 = \langle 1 \rangle \oplus K$$

$$
\begin{aligned}
x_0 \oplus x_1 &= (\langle 0 \rangle \oplus k) \oplus (\langle 1 \rangle \oplus k) \\
&= (\langle 0 \rangle \oplus \langle 1 \rangle) \oplus (k \oplus k) \qquad \text{XOR is communitive} \\
&= \langle 1 \rangle \oplus \langle 0 \rangle \\
&= \langle 1 \rangle
\end{aligned}
$$

$$
\begin{aligned}
\text{Advantage} &= \Pr[\text{output perm384} \mid f \text{ is perm384}] \\
&\quad - \Pr[\text{output perm384} \mid f \text{ is rand perm}] \\
&= 1 - \frac{1}{2^{384}} \\
&\approx 1
\end{aligned}
$$

Try 3



$$f(x) = k \oplus \text{perm384}(k \oplus x)$$

when k is random and secret, f is indistinguishable from
a fresh random permutation

A block cipher is an algorithm that is indistinguishable from a random permutation when given a random key.

- most widely used

	Key size	Block size
• DES (1970s)	56 bits	64 bits
• AES (1990s)	128 192 bit 256	128 bits

- DES: Data Encryption Standard
- AES:
  - Advanced Encryption Standard
  - built into PCs and phones

AES w/ 128 bit key AES128
" " 256 " " AES256