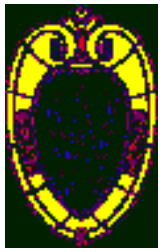

CS/ECE/Math 435
Introduction to Cryptography

Professor Chris DeMarco
Department of Electrical & Computer Engineering
University of Wisconsin-Madison

Spring Semester 2021

Goal of understanding DES, the Data Encryption Standard.

However, like many standards, DES includes many specific details that reflect somewhat arbitrary (or dated) design choices, made at time the standard was set.

It is useful to be aware of all these details, but main goal in 435 emphasizes core underlying concepts behind DES, that remain most useful today.

Approach today (slight variation on treatment in sections 26 & 27 of Bach notes):

We'll first describe a somewhat idealized nonlinear block encryption scheme <u>that</u> <u>is</u> <u>not</u> <u>DES</u>.

However, our scheme will include several of the core constructs of DES, hopefully illustrated in a cleaner, clearer fashion.

First core construct: "Feistel Ciphers"

- A specific means of constructing an <u>invertible</u> (typically nonlinear) function from an arbitrary (not necessarily invertible) function.

Recall our notation: integer m here denotes block size for a block cipher, l denotes key length.

So we seek to construct functions

$$B: \underbrace{Z_2^m}_{\substack{m\text{-sized} \\ \text{block} \\ \text{plaintext}}} \times \underbrace{Z_2^l}_{\substack{l\text{-sized} \\ \text{binary} \\ \text{key}}} \to \underbrace{Z_2^m}_{\substack{m\text{-sized} \\ \text{block ciphertext}}}$$

We'll sometimes write as

$$\underline{z} = B(\underline{w}, \underline{k}) \quad \text{OR} \quad z = B_{\underline{k}}(\underline{w})$$

$\underline{z}$: m-dimensional binary vectors

$\underline{w}$: m-dimensional binary vectors

$\underline{k}$: ℓ dimensional binary vector

Typically require that for all $\hat{\underline{k}} \in \mathbb{Z}_2^{\ell}$

$B_{\hat{\underline{k}}} : \mathbb{Z}_2^{m} \to \mathbb{Z}_2^{m}$ is a 1-1 function, with well defined functional inverse $B^{-1}_{\hat{\underline{k}}}(\cdot)$, i.e. for all $\underline{w} \in \mathbb{Z}_2^{m}$

$$B^{-1}_{\hat{\underline{k}}}\left(B_{\hat{\underline{k}}}(\underline{w})\right) = \underline{w}$$

Simple but clever idea due to Horst Feistel (IBM researcher of 50's & 60's):

Without loss of generality, assume m is even (we can always "pad out" to one extra component if m is odd).

Let $b : \mathbb{Z}_2^{m/2} \times \mathbb{Z}_2^{\ell} \to \mathbb{Z}_2^{m/2}$

write $b(\underline{y}, \underline{k})$ or $b_{\underline{k}}(\underline{y})$

$\underline{y}$: $m/2$ dimensional binary vector

NO restrictions on "b" any function acceptable!

(Note: Bach notes use "f" for function I label as "b" above).

Then, for given $\underline{w} \in \mathbb{Z}_2^m$

(typically, $\underline{w}$ is an m-sized block of plaintext), partition $\underline{w}$ as

$$\underline{w} = \begin{bmatrix} \underline{x} \\ \hline \underline{y} \end{bmatrix} \begin{matrix} \} \text{ size } m/2 \\ \} \text{ size } m/2 \end{matrix}$$

Then, define our $B_{\underline{k}}(\underline{w})$ as

$$B_k(\underline{w}) = \begin{bmatrix} \underline{y} \\ \hline \underline{x} + b_k(\underline{y}) \end{bmatrix}$$

Bach notes illustrate with trivial choices of b_k functions (constant, linear, affine).

Let's consider more useful classes of $b(\underline{y}, \underline{k})$ or $b_{\underline{k}}(\underline{y})$.

Because we're operating on vectors with components $\{0, 1\}$, one can view $b(\underline{y}, \underline{k})$ functions as Boolean functions, or collections of logical "sentences." But for cryptographic perspective, better to keep algebraic viewpoint.

We allow $b(\underline{y}, \underline{k})$, componentwise, to be multivariable polynomial in y_i's and k_i's. For example,

$$\underbrace{\underline{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}}_{m/2 = 2}, \quad \underbrace{k = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}}_{\ell = 2}$$

a suitable $b(\underline{y}, \underline{k})$ might be

$$\left.\begin{bmatrix} y_1 \cdot k_2 \cdot y_2 + k_1 \cdot y_2 \\ \hline y_1 \cdot k_1 \cdot y_2 + k_2 \cdot y_1 \end{bmatrix}\right\} \text{modulo 2 of course}$$

Q: Is B_k^{-1} guaranteed to exist, and if so, how do we compute it?

A: (Feistel's very clever insight):

Suppose

$$z = \begin{bmatrix} x' \\ \hline y' \end{bmatrix} = B_k(w) = B_k\left(\begin{bmatrix} x \\ \hline y \end{bmatrix}\right) = \begin{bmatrix} y \\ \hline x + b_k(y) \end{bmatrix}$$

then we claim:

$$B_k^{-1}(z) = \begin{bmatrix} y' - b_k(x') \\ \hline x' \end{bmatrix}.$$

Proof by direct calculation:

Evaluate $\underline{z} = B_{\underline{k}}\left(\begin{bmatrix} \underline{x} \\ \hline \underline{y} \end{bmatrix}\right) \Rightarrow \underline{z} = \begin{bmatrix} \underline{x}' \\ \hline \underline{y}' \end{bmatrix} = \begin{bmatrix} \underline{y} \\ \hline \underline{x} + b_k(\underline{y}) \end{bmatrix}$

Then $B_k^{-1}\left(\begin{bmatrix} \underline{x}' \\ \hline \underline{y}' \end{bmatrix}\right) = B_k^{-1}\left(\begin{bmatrix} \underline{y} \\ \hline \underline{x} + b_k(\underline{y}) \end{bmatrix}\right)$

$$= \begin{bmatrix} \overbrace{\{\underline{x} + b_k(\underline{y})\}}^{\text{this is } y'} - b_k(\overbrace{\underline{y}}^{\text{this is } x'}) \\ \hline \underbrace{\underline{y}}_{\text{this is } \underline{x}'} \end{bmatrix} = \begin{bmatrix} \underline{x} \\ \hline \underline{y} \end{bmatrix}$$

Consider our example b_k for specific values

$$\underline{w} = [1\ 0\ 0\ 1]^T, \quad \underline{k} = [1\ 1]^T$$

Partition $\underline{w}$: $\underline{x} = [1\ 0]^T$, $\underline{y} = [0\ 1]^T$

$$B_k(\underline{w}) = \underbrace{\begin{bmatrix} 0 \\ 1 \\ \hline \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0\cdot 1\cdot 1 + 1\cdot 1 \\ 0\cdot 1\cdot 1 + 1\cdot 0 \end{bmatrix} \end{bmatrix}}_{\text{evaluate } b_{\underline{k}}(\underline{y})} \bmod 2 = \underbrace{\begin{bmatrix} 0 \\ 1 \\ \hline 0 \\ 0 \end{bmatrix}}_{\text{resulting ciphertext block } \underline{z}}$$

Just to illustrate, also calculate $B_k^{-1}(\underline{z})$. Observe $\underline{x}' = [0\ 1]^T$; $\underline{y}' = [0\ 0]^T$

NOTE: apply to $\underline{x}'$ here!

$$B_k^{-1}(\underline{z}) = \left[\frac{\underline{y}' - b_{\underline{k}}(\underline{x}')}{\underline{x}'} \right] = \left[\frac{\begin{bmatrix} 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \cdot 1 \cdot 1 + 1 \cdot 1 \\ 0 \cdot 1 \cdot 1 + 1 \cdot 0 \end{bmatrix}}{\begin{bmatrix} 0 \\ 1 \end{bmatrix}} \right] \bmod 2$$

$$= \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \text{original } \underline{w}, \text{ as desired.}$$

Question: We'd like our $B_{\underline{k}}(\underline{w})$ functions to be nonlinear with respect to $\underline{w}$. Doesn't the Feistel construction force us to a $B_k(\underline{w})$ that is "half linear" with respect to $\underline{w}$? (i.e. in the portion, half the output $\underline{z}$ is just reordered elements of $\underline{w}$).

Ans: Yes, ... but easy fix.

Improving nonlinearity of Feistel construction:

- Primary fix: compose multiple instances of $B_{\underline{k}}(\underline{w})$, often with changes to $\underline{k}$ (we'll address systematic, "DES-like" methods to generate multiple "subkeys" from a "master key" shortly).

So, simply let

$$B_{\underline{k}}^{\text{overall}}(\underline{w}) = B_{\underline{k}_2}\left(B_{\underline{k}_1}(\underline{w})\right)$$

$\underline{k}$: master key

subkeys $\underline{k}_1, \underline{k}_2$ derived from master key

In terminology of Feistel, composing multiple instances of function is termed "rounds of Feistel." Example above was two rounds.

DES standard applies 16 rounds of Feistel (i.e. it composes 16 instances of $B_{k_i}(\underline{w})$, with distinct k_i subkeys derived from overall $\ell = 56$ (56 bit) master key.

- Secondary fix: introduce additional simple linear operations (of non-Feistel form) between Feistel-based $B_k(\underline{w})$ functions. Most typically, these are simple permutations. (DES uses a specific 64-bit permutation, labeled as the "σ-permutation": see Bach notes p. 6-4 for definition of this σ-permutation).

Additional Operations in preparation for details of actual DES:

- In 435, we like our $b_{\underline{k}}(\circ)$'s explicitly defined as closed-form algebraic functions on $\mathbb{Z}_2^m$. But for modest dimensioned m, speed of evaluation historically favored look-up table approach to function definition (Caution / Update: in many modern applications and architectures, we're often more concerned with watts/operation, less with time/operation. Look-up table function evaluation can become less attractive)

- In defining its "Feistel Rounds" (the $b_{\underline{k}_i}(\cdot)$'s in notation of today's lecture), DES does use look-up table approach.

Consider our $b_{\underline{k}}(\cdot)$ from p. (8)

$$\begin{bmatrix} y_1 \cdot k_2 \cdot y_2 + k_1 \cdot y_2 \\ y_1 \cdot k_1 \cdot y_2 + k_2 \cdot y_1 \end{bmatrix}, \quad \text{for } \underline{k} = [1 \;\; 1]^T$$

Look-up Table

$\underline{y}$	$b_{\underline{k}}(\underline{y})$
$[0 \;\; 0]^T$	$[0 \;\; 0]^T$
$[0 \;\; 1]^T$	$[1 \;\; 0]^T$
$[1 \;\; 0]^T$	$[0 \;\; 1]^T$
$[1 \;\; 1]^T$	$[0 \;\; 0]^T$

Aside: note that $b_{\underline{k}}(\cdot)$ is not 1-1, not invertible. That's ok ... associated $B_{\underline{k}}(\underline{w})$ is guaranteed invertible.

So... many of the (IMO, tedious) details of the DES standard can be placed in framework described in this lecture:

1) From DES's 56-bit ($\ell = 56$) master key, how are sub-keys constructed?

2) Using these subkeys, how are each of the 16 $b_{k_i}(\cdot)$ functions (with their associated, "larger" $B_{k_i}(\cdot)$'s) constructed to create DES's 16 "Feistel Rounds."