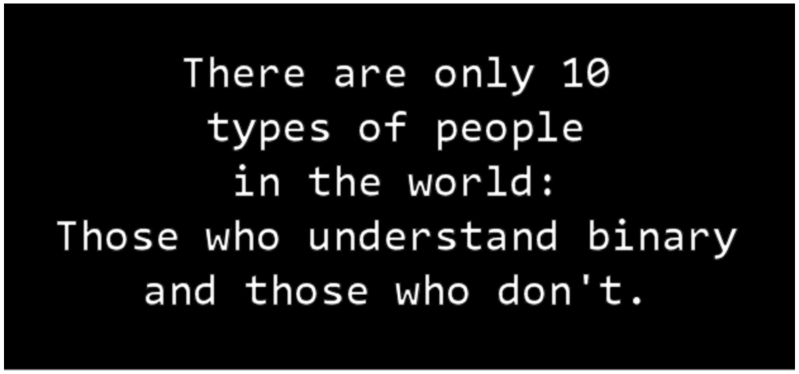
BLOCK CIPHERS
and
KEY-RECOVERY SECURITY

Notation


There are only 10 types of people in the world: Those who understand binary and those who don't.

Notation

$\{0,1\}^n$ is the set of n-bit strings and $\{0,1\}^*$ is the set of all strings of finite length. By ε we denote the empty string.

If S is a set then $|S|$ denotes its size. Example: $|\{0,1\}^2| = 4$.

If x is a string then $|x|$ denotes its length. Example: $|0100| = 4$.

think this as a possible of all remainders when divided by m

If $m \geq 1$ is an integer then let $\mathbf{Z}_m = \{0, 1, \ldots, m-1\}$.

By $x \xleftarrow{\$} S$ we denote picking an element at random from set S and assign it to x. Thus $\Pr[x = s] = 1/|S|$ for every $s \in S$.

Functions

f is the function which takes X1, ..Xn as inputs and gives output Y.

Let $n \geq 1$ be an integer. Let $X_1, \ldots, X_n$ and Y be (non-empty) sets.

By $f\colon X_1 \times \cdots \times X_n \to Y$ we denote that f is a function that

- Takes inputs $x_1, \ldots, x_n$, where $x_i \in X_i$ for $1 \leq i \leq n$
- and returns an output $y = f(x_1, \ldots, x_n) \in Y$. y is one of the value from set Y

We call n the number of inputs (or arguments) of f. We call $X_1 \times \cdots \times X_n$ the domain of f and Y the range of f.

{0,1} {0, 1,2}

Example: Define $f\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ by $f(x_1, x_2) = (x_1 + x_2) \bmod 3$. This is a function with $n = 2$ inputs, domain $\mathbf{Z}_2 \times \mathbf{Z}_3$ and range $\mathbf{Z}_3$.

Permutations

* permutation is a particular type of function in cryptography. It moves us closer to encryption by being uniquely invertible which when we use it with decryption

Suppose $f\colon X \to Y$ is a function with one argument. We say that it is a *permutation* if

- $X = Y$, meaning its domain and range are the same set.
- There is an *inverse* function $f^{-1}\colon Y \to X$ such that $f^{-1}(f(x)) = x$ for all $x \in X$. Structurely,the function is bijection. i.e its one to one and onto

This means f must be one-to-one and onto: for every $y \in Y$ there is a unique $x \in X$ such that $f(x) = y$.

Permutations versus functions example

Consider the following two functions $f\colon \{0,1\}^2 \to \{0,1\}^2$, where $X = Y = \{0,1\}^2$:

x	00	01	10	11
$f(x)$	01	11	00	10

A permutation

x	00	01	10	11
$f(x)$	01	11	11	10

Not a permutation

Two different input maps to the same output.. so the function is not invertible

Permutations versus functions example

Consider the following two functions $f\colon \{0,1\}^2 \to \{0,1\}^2$, where $X = Y = \{0,1\}^2$:

x	00	01	10	11
$f(x)$	01	11	00	10

A permutation

x	00	01	10	11
$f(x)$	01	11	11	10

Not a permutation

x	00	01	10	11
$f^{-1}(x)$	10	00	11	01

Its inverse

Function families we want to introduce an idea of key in family of functions

everytime you have a key (a particular or different choice of key), you will induce a new function. The way we capture that, is by defining a family of functions

A family of functions (also called a function family) is a two-input function F : Keys $\times$ D $\to$ R. For $K \in$ Keys we let F_K : D $\to$ R be defined by $F_K(x) = F(K, x)$ for all $x \in$ D. when we fix the key K, we induce a function from D to R

- The set Keys is called the key space. If Keys $= \{0,1\}^k$ we call k the key length.
- The set D is called the input space. If D $= \{0,1\}^\ell$ we call ℓ the input length.
- The set R is called the output space or range. If R $= \{0,1\}^L$ we call L the output length.

Example: Define $F\colon \underset{\text{[key space]}}{\overset{\{0,1\}}{\mathbf{Z}_2}} \times \underset{\text{[input space]}}{\overset{\{0,1,2\}}{\mathbf{Z}_3}} \to \mathbf{Z}_3$ by $F(K,x) = (K \cdot x) \bmod 3$.

- This is a family of functions with domain $\mathbf{Z}_2 \times \mathbf{Z}_3$ and range $\mathbf{Z}_3$.
- If $K = 1$ then $F_K\colon \mathbf{Z}_3 \to \mathbf{Z}_3$ is given by $F_K(x) = x \bmod 3$. {we fixed the key}

Block ciphers: Definition

Block cipher is specific or particular type of family of functions. It has a key space input space and range space.

Let $E\colon \text{Keys} \times \text{D} \to \text{R}$ be a family of functions. We say that E is a block cipher if

- $\text{R} = \text{D}$, meaning the input and output spaces are the same set.
- $E_K : \text{D} \to \text{D}$ is a permutation for every key $K \in \text{Keys}$, meaning has an inverse $E_K^{-1} : \text{D} \to \text{D}$ such that $E_K^{-1}(E_K(x)) = x$ for all $x \in \text{D}$.

We let $E^{-1} : \text{Keys} \times \text{D} \to \text{D}$, defined by $E^{-1}(K, y) = E_K^{-1}(y)$, be the inverse block cipher to E.

In practice we want that E, E^{-1} are efficiently computable.

If $\text{Keys} = \{0,1\}^k$ then k is the key length as before. If $\text{D} = \{0,1\}^\ell$ we call ℓ the block length.

Block ciphers: Example

Block cipher $E\colon \{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ (left), where the table entry corresponding to the key in row K and input in column x is $E_K(x)$. Its inverse $E^{-1}\colon \{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ (right).

{inputs}

{keys}

	00	01	10	11
00	11	00	10	01
01	11	10	01	00
10	10	11	00	01
11	11	00	10	01

{outputs}

{keys}

	00	01	10	11
00	01	11	10	00
01	11	10	01	00
10	10	11	00	01
11	01	11	10	00

- Row 01 of E equals Row 01 of E^{-1}, meaning $E_{01} = E_{01}^{-1}$
- Rows have no repeated entries, for both E and E^{-1}
- Column 00 of E has repeated entries, that's ok
- Rows 00 and 11 of E are the same, that's ok

In practice, you can't write these block ciphers in table as shown above, instead you write in code

Block Ciphers: Example

Block cipher specified in codes

Let $\ell = k$ and define $E\colon \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^\ell$ by

$$E_K(x) = E(K, x) = K \oplus x$$

Then E_K has inverse E_K^{-1} where

$$E_K^{-1}(y) = K \oplus y$$

Why? Because

$$E_K^{-1}(E_K(x)) = E_K^{-1}(K \oplus x) = K \oplus K \oplus x = x$$

The inverse of block cipher E is the block cipher E^{-1} defined by

$$E^{-1}(K, y) = E_K^{-1}(y) = K \oplus y$$

Exercise

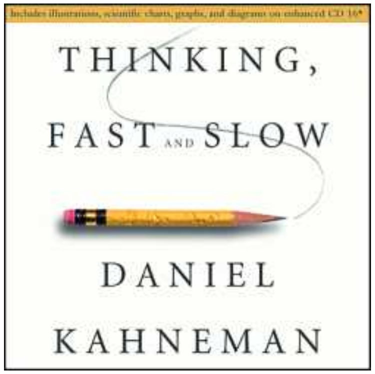
Let $E\colon \text{Keys} \times \text{D} \to \text{D}$ be a block cipher. Is E a permutation?

- YES
- NO
- QUESTION DOESN'T MAKE SENSE
- WHO CARES?

This is an exercise in correct mathematical language.

The way the permutation is defined is it has a single input but block cipher has two inputs. So the answer is NO

Slow is good


Includes illustrations, scientific charts, graphs, and diagrams on enhanced CD 16*
THINKING,
FAST AND SLOW
DANIEL
KAHNEMAN

Exercise

Let $E\colon \text{Keys} \times \text{D} \to \text{D}$ be a block cipher. Is E a permutation?

How to proceed to answer this: Think slow. Don't jump to a conclusion. Instead:

- Look back at the definition of a block cipher.
- Look back at the definition of a permutation.
- Pattern match these.
- Now make an informed and justified conclusion.

This is an exercise in correct mathematical language.

This is considered a *high-school level* exercise.

Exercise

Above we had given the following example of a family of functions:
$F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ defined by $F(K, x) = (K \cdot x) \bmod 3$.

Question: Is F a block cipher? Why or why not?

```
  If you fix K, D->D.. so input and output space are the same
  For every choice of k in K, Fk should be permutation.
  * that means, it should be intertible for every choice of k

Key space has two possible values- {0, 1}
=> we can fix K to 0: then, F0(x) = 0 mod 3 = 0 (always) ==> Is this Invertible??
* for any possible value of input x, the output will always be 0, which is many-to-one
  and so it is not invertible and hence not purmutation.
==> fixing K to 1: F1(x) = x mod 3 ==> 0--> 0, 1-->1, 2-->2. So its one to one function
  and hnce intertible
```

Exercise

Above we had given the following example of a family of functions:
$F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ defined by $F(K, x) = (K \cdot x) \bmod 3$.

Question: Is F a block cipher? Why or why not?

Answer: No, because $F_0(1) = F_0(2)$ so F_0 is not a permutation.

Exercise

Above we had given the following example of a family of functions: $F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ defined by $F(K,x) = (K \cdot x) \bmod 3$.

Question: Is F a block cipher? Why or why not?

Answer: No, because $F_0(1) = F_0(2)$ so F_0 is not a permutation.

Question: Is F_1 a permutation?

Exercise

Above we had given the following example of a family of functions: $F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ defined by $F(K,x) = (K \cdot x) \bmod 3$.

Question: Is F a block cipher? Why or why not?

Answer: No, because $F_0(1) = F_0(2)$ so F_0 is not a permutation.

Question: Is F_1 a permutation?

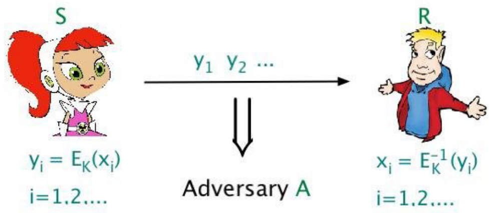
Answer: Yes. But that alone does not make F a block cipher.

Block cipher usage

{key space} {input space} {output space}

Let $E\colon \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^\ell$ be a block cipher. It is considered public. In typical usage

- $K \xleftarrow{\$} \{0,1\}^k$ is known to parties S, R, but not given to adversary A.
  K is chosen randomly from the key space
- S, R use E_K for encryption



Leads to security requirements like: Hard to get K from $y_1, y_2, \ldots$; Hard to get x_i from y_i; ...

DES History

{Data Encryption standard}

1972 – NBS (now NIST) asked for a block cipher for standardization

1974 – IBM designs Lucifer

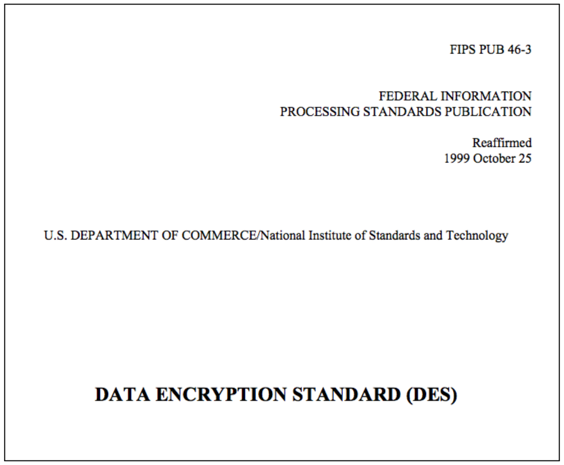
Lucifer eventually evolved into DES.

Widely adopted as a standard including by ANSI and American Bankers association

Used in ATM machines

Replaced (by AES) in 2001.

FIPS DES Standard: Reaffirmed 1999



FIPS PUB 46-3

FEDERAL INFORMATION
PROCESSING STANDARDS PUBLICATION

Reaffirmed
1999 October 25

U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology

DATA ENCRYPTION STANDARD (DES)

DES parameters

Note that for block cipher, input space and output space has the same length, but key can be of any length.

Key Length $k = 56$

Block length $\ell = 64$

So,

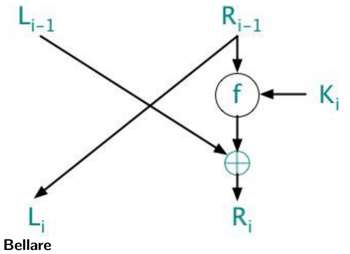
$$\text{DES}\colon \{0,1\}^{56} \times \{0,1\}^{64} \to \{0,1\}^{64}$$

$$\text{DES}^{-1}\colon \{0,1\}^{56} \times \{0,1\}^{64} \to \{0,1\}^{64}$$

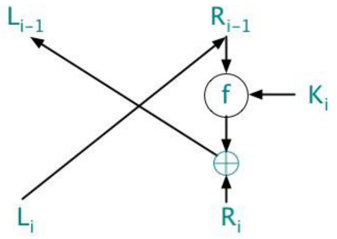
DES Construction

```
function $\text{DES}_K(M)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \le i \le 16$
    $M \leftarrow IP(M)$
    Parse $M$ as $L_0 \| R_0$    // $|L_0| = |R_0| = 32$
    for $i = 1$ to 16 do
        $L_i \leftarrow R_{i-1}$ ;   $R_i \leftarrow f(K_i, R_{i-1}) \oplus L_{i-1}$
    $C \leftarrow IP^{-1}(L_{16} \| R_{16})$
    return $C$
```

Round i:



Invertible given K_i:



DES Construction

```
function $\text{DES}_K(M)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \le i \le 16$
    $M \leftarrow IP(M)$
    Parse $M$ as $L_0 \| R_0$    // $|L_0| = |R_0| = 32$
    for $i = 1$ to 16 do
        $L_i \leftarrow R_{i-1}$ ;   $R_i \leftarrow f(K_i, R_{i-1}) \oplus L_{i-1}$
    $C \leftarrow IP^{-1}(L_{16} \| R_{16})$
    return $C$

function $\text{DES}_K^{-1}(C)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \le i \le 16$
    $C \leftarrow IP(C)$
    Parse $C$ as $L_{16} \| R_{16}$
    for $i = 16$ downto 1 do
        $R_{i-1} \leftarrow L_i$ ;   $L_{i-1} \leftarrow f(K_i, R_{i-1}) \oplus R_i$
    $M \leftarrow IP^{-1}(L_0 \| R_0)$
    return $M$
```

DES Construction

```
function $\text{DES}_K(M)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \le i \le 16$
    $M \leftarrow IP(M)$
    Parse $M$ as $L_0 \,\|\, R_0$    // $|L_0| = |R_0| = 32$
    for $i = 1$ to 16 do
        $L_i \leftarrow R_{i-1}$ ;    $R_i \leftarrow f(K_i, R_{i-1}) \oplus L_{i-1}$
    $C \leftarrow IP^{-1}(L_{16} \,\|\, R_{16})$
    return $C$
```

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

DES Construction

```
function $f(J, R)$    // $|J| = 48$ and $|R| = 32$
    $R \leftarrow E(R)$ ;    $R \leftarrow R \oplus J$
    Parse $R$ as $R_1 \,\|\, R_2 \,\|\, R_3 \,\|\, R_4 \,\|\, R_5 \,\|\, R_6 \,\|\, R_7 \,\|\, R_8$    // $|R_i| = 6$ for $1 \le i \le$
    for $i = 1, \ldots, 8$ do
        $R_i \leftarrow \mathbf{S}_i(R_i)$    // Each S-box returns 4 bits
    $R \leftarrow R_1 \,\|\, R_2 \,\|\, R_3 \,\|\, R_4 \,\|\, R_5 \,\|\, R_6 \,\|\, R_7 \,\|\, R_8$    // $|R| = 32$ bits
    $R \leftarrow P(R)$ ; return $R$
```

E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

S-boxes

			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
$\mathbf{S}_1$:	0	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	1	0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	1	1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
$\mathbf{S}_2$:	0	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	1	0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	1	1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

			0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
$\mathbf{S}_3$:	0	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	1	0	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1	1	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Key Recovery Attack Scenario

Let E: Keys $\times$ D $\rightarrow$ R be a block cipher known to the adversary A.

- Sender Alice and receiver Bob share a *target key* $K \in$ Keys.
- Alice encrypts M_i to get $C_i = E_K(M_i)$ for $1 \le i \le q$, and transmits $C_1, \ldots, C_q$ to Bob
- The adversary gets $C_1, \ldots, C_q$ and also knows $M_1, \ldots, M_q$
- Now the adversary wants to figure out K so that it can decrypt any future ciphertext C to recover $M = E_K^{-1}(C)$.

Question: Why do we assume A knows $M_1, \ldots, M_q$?

Answer: Reasons include a posteriori revelation of data, a priori knowledge of context, and just being conservative!

Key Recovery Security Metrics

We consider two measures (metrics) for how well the adversary does at this key recovery task:

- Target key recovery (TKR)
- Consistent key recovery (KR)

In each case the definition involves a game and an advantage.

The definitions will allow E to be any family of functions, not just a block cipher.

The definitions allow A to pick, not just know, $M_1, \ldots, M_q$. This is called a chosen-plaintext attack.

Target Key Recovery Definitions: Game and Advantage

```
Game $\mathrm{TKR}_E$
procedure Initialize
$K \xleftarrow{\$} \mathrm{Keys}$

procedure Fn($M$)
Return $E(K, M)$

procedure Finalize($K'$)
Return $(K = K')$
```

Definition: $\mathbf{Adv}_E^{\mathrm{tkr}}(A) = \Pr[\mathrm{TKR}_E^A \Rightarrow \mathrm{true}]$.

- First **Initialize** executes, selecting *target key* $K \xleftarrow{\$} \mathrm{Keys}$, but not giving it to A.
- Now A can call (query) **Fn** on any input $M \in \mathrm{D}$ of its choice to get back $C = E_K(M)$. It can make as many queries as it wants.
- Eventually A will halt with an output K' which is automatically viewed as the input to **Finalize**
- The game returns whatever **Finalize** returns
- The tkr advantage of A is the probability that the game returns true

Consistent keys

Def: Let E: Keys $\times$ D $\rightarrow$ R be a family of functions. We say that key $K' \in$ Keys is *consistent* with $(M_1, C_1), \ldots, (M_q, C_q)$ if $E(K', M_i) = C_i$ for all $1 \leq i \leq q$.

Example: For E: $\{0,1\}^2 \times \{0,1\}^2 \rightarrow \{0,1\}^2$ defined by

	00	01	10	11
00	11	00	10	01
01	11	10	01	00
10	10	11	00	01
11	11	00	10	01

The entry in row K, column M is $E(K, M)$.

- Key 00 is consistent with $(11, 01)$
- Key 10 is consistent with $(11, 01)$
- Key 00 is consistent with $(01, 00), (11, 01)$
- Key 11 is consistent with $(01, 00), (11, 01)$

Consistent Key Recovery Definitions: Game and Advantage

Let E: Keys $\times$ D $\rightarrow$ R be a family of functions, and A an adversary.

```
Game $\mathrm{KR}_E$
procedure Initialize
$K \xleftarrow{\$} \mathrm{Keys}$; $i \leftarrow 0$

procedure Fn($M$)
$i \leftarrow i + 1$; $M_i \leftarrow M$
$C_i \leftarrow E(K, M_i)$
Return $C_i$

procedure Finalize($K'$)
$\mathrm{win} \leftarrow$ true
For $j = 1, \ldots, i$ do
    If $E(K', M_j) \neq C_j$ then $\mathrm{win} \leftarrow$ false
    If $M_j \in \{M_1, \ldots, M_{j-1}\}$ then $\mathrm{win} \leftarrow$ false
Return $\mathrm{win}$
```

Definition: $\mathbf{Adv}_E^{\mathrm{kr}}(A) = \Pr[\mathrm{KR}_E^A \Rightarrow \mathrm{true}]$.

The game returns true if (1) The key K' returned by the adversary is consistent with $(M_1, C_1), \ldots, (M_q, C_q)$, and (2) $M_1, \ldots, M_q$ are distinct.

A is a q-query adversary if it makes q distinct queries to its **Fn** oracle.

kr advantage always exceeds tkr advantage

Fact: Suppose that, in game KR_E, adversary A makes queries $M_1, \ldots, M_q$ to **Fn**, thereby defining $C_1, \ldots, C_q$. Then the target key K is consistent with $(M_1, C_1), \ldots, (M_q, C_q)$.

Proposition: Let E be a family of functions. Let A be *any* adversary all of whose **Fn** queries are distinct. Then

$$\mathbf{Adv}^{\mathrm{kr}}_E(A) \geq \mathbf{Adv}^{\mathrm{tkr}}_E(A) .$$

Why? If the K' that A returns equals the target key K, then, by the Fact, the input-output examples $(M_1, C_1), \ldots, (M_q, C_q)$ will of course be consistent with K'.

Exhaustive Key Search attack

Let E: Keys $\times$ D $\rightarrow$ R be a function family with Keys $= \{T_1, \ldots, T_N\}$ and D $= \{x_1, \ldots, x_d\}$. Let $1 \leq q \leq d$ be a parameter.

adversary A_{eks}

```
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

Question: What is $\mathbf{Adv}^{\mathrm{kr}}_E(A_{\mathrm{eks}})$?

Exhaustive Key Search attack

Let E: Keys $\times$ D $\rightarrow$ R be a function family with Keys $= \{T_1, \ldots, T_N\}$ and D $= \{x_1, \ldots, x_d\}$. Let $1 \leq q \leq d$ be a parameter.

adversary A_{eks}

```
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

Question: What is $\mathbf{Adv}^{\mathrm{kr}}_E(A_{\mathrm{eks}})$?

Answer: It equals 1.

Because

- There is some i such that $T_i = K$, and
- K is consistent with $(M_1, C_1), \ldots, (M_q, C_q)$.

Exhaustive Key Search attack

Let E: Keys $\times$ D $\rightarrow$ R be a function family with Keys $= \{T_1, \ldots, T_N\}$ and D $= \{x_1, \ldots, x_d\}$. Let $1 \leq q \leq d$ be a parameter.

adversary A_{eks}

```
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

Question: What is $\mathbf{Adv}^{\mathrm{tkr}}_E(A_{\mathrm{eks}})$?

Exhaustive Key Search attack

Let E: Keys × D → R be a function family with Keys $= \{T_1, \ldots, T_N\}$ and D $= \{x_1, \ldots, x_d\}$. Let $1 \le q \le d$ be a parameter.

adversary A_{eks}

```
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

Question: What is $\mathbf{Adv}_E^{\text{tkr}}(A_{\text{eks}})$?

Answer: Hard to say! Say $K = T_m$ but there is a $i < m$ such that $E(T_i, M_j) = C_j$ for $1 \le j \le q$. Then T_i, rather than K, is returned.

In practice if $E \colon \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^\ell$ is a "real" block cipher and $q > k/\ell$, we expect that $\mathbf{Adv}_E^{\text{tkr}}(A_{\text{eks}})$ is close to 1 because K is likely the only key consistent with the input-output examples.

Exercise: tkr advantage can be much less than kr

Let $k, \ell \ge 1$ be given integers. Present in pseudocode a block cipher $E \colon \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^\ell$ for which you do the following:

(1) Given any positive integer $q \le 2^\ell$, present in pseudocode a q-query, $\mathcal{O}(q(k+\ell))$-time adversary A_q with $\mathbf{Adv}_E^{\text{kr}}(A_q) = 1$.

(2) Prove that $\mathbf{Adv}_E^{\text{tkr}}(A) \le 2^{-k}$ for any adversary A.

So far: A Pedagogic interlude

- Slides 1–18 are basic mathematical notation and definitions at a CSE 20 level. You should find this easy.
- Slides 19–27 (DES) are just a story. Don't congratulate yourself if you "understand" it because there really isn't anything to understand, at least at the level we told the story.
- Slides 28–38 are representative of what you need to understand to do well. There is depth here. It takes time, thought and many passes to understand.

How long does exhaustive key search take?

DES can be computed at 1.6 Gbits/sec in hardware.

DES plaintext = 64 bits

Chip can perform $(1.6 \times 10^9)/64 = 2.5 \times 10^7$ DES computations per second

Expect A_{eks} ($q = 1$) to succeed in 2^{55} DES computations, so it takes time

$$\begin{aligned} \frac{2^{55}}{2.5 \times 10^7} &\approx 1.4 \times 10^9 \text{ seconds} \\ &\approx 45 \text{ years!} \end{aligned}$$

Key Complementation ⇒ 22.5 years

But this is prohibitive. Does this mean DES is secure?

Differential and linear cryptanalysis

Exhaustive key search is a generic attack: Did not attempt to "look inside" DES and find/exploit weaknesses.

The following non-generic key-recovery attacks on DES have advantage close to one and running time smaller than 2^{56} DES computations:

Attack	when	q, running time
Differential cryptanalysis	1992	2^{47}
Linear cryptanalysis	1993	2^{44}

Differential and linear cryptanalysis

Exhaustive key search is a generic attack: Did not attempt to "look inside" DES and find/exploit weaknesses.

The following non-generic key-recovery attacks on DES have advantage close to one and running time smaller than 2^{56} DES computations:

Attack	when	q, running time
Differential cryptanalysis	1992	2^{47}
Linear cryptanalysis	1993	2^{44}

But merely storing 2^{44} input-output pairs requires 281 Tera-bytes.

In practice these attacks were prohibitively expensive.

EKS revisited

```
adversary $A_{\mathrm{eks}}$
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

EKS revisited

```
adversary $A_{\mathrm{eks}}$
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
```

Observation: The E computations can be performed in parallel!

EKS revisited

```
adversary $A_{\text{eks}}$
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
For $i = 1, \ldots, N$ do
    if $(\forall j \in \{1, \ldots, q\} \; : \; E(T_i, M_j) = C_j)$ then return $T_i$
```

Observation: The E computations can be performed in parallel!

In 1993, Wiener designed a dedicated DES-cracking machine:

- \$1 million
- 57 chips, each with many, many DES processors
- Finds key in 3.5 hours

RSA DES challenges

$K \xleftarrow{\$} \{0,1\}^{56}$; $Y \leftarrow \mathrm{DES}(K, X)$; Publish Y on website.
Reward for recovering X

Challenge	Post Date	Reward	Result
I	1997	\$10,000	Distributed.Net: 4 months
II	1998	Depends how fast you find key	Distributed.Net: 41 days. EFF: 56 hours
III	1998	As above	$<$ 28 hours

DES security summary

DES is considered broken because its short key size permits rapid key-search.

But DES is a very strong design as evidenced by the fact that there are no practical attacks that exploit its structure.

2DES

Block cipher $2DES : \{0,1\}^{112} \times \{0,1\}^{64} \rightarrow \{0,1\}^{64}$ is defined by

$$2DES_{K_1K_2}(M) = DES_{K_2}(DES_{K_1}(M))$$

- Exhaustive key search takes 2^{112} *DES* computations, which is too much even for machines
- Resistant to differential and linear cryptanalysis.

Meet-in-the-middle attack on 2DES

Suppose K_1K_2 is a target 2DES key and adversary has M, C such that

$$C = 2DES_{K_1K_2}(M) = DES_{K_2}(DES_{K_1}(M))$$

Then

$$DES^{-1}_{K_2}(C) = DES_{K_1}(M)$$

Meet-in-the-middle attack on 2DES

Suppose $DES^{-1}_{K_2}(C) = DES_{K_1}(M)$ and $T_1, \ldots, T_N$ are all possible DES keys, where $N = 2^{56}$.

T_1	$DES(T_1, M)$
T_i	$DES(T_i, M)$
T_N	$DES(T_N, M)$

Table L

$DES^{-1}(T_1, C)$	T_1
$DES^{-1}(T_j, C)$	T_j
$DES^{-1}(T_N, C)$	T_N

Table R

Attack idea:

- Build L,R tables

Meet-in-the-middle attack on 2DES

Suppose $DES^{-1}_{K_2}(C) = DES_{K_1}(M)$ and $T_1, \ldots, T_N$ are all possible DES keys, where $N = 2^{56}$.

$K_1 \rightarrow$

T_1	$DES(T_1, M)$
T_i	$DES(T_i, M)$
T_N	$DES(T_N, M)$

Table L

$\overset{\text{equal}}{\longleftrightarrow}$

$DES^{-1}(T_1, C)$	T_1
$DES^{-1}(T_j, C)$	T_j
$DES^{-1}(T_N, C)$	T_N

$\leftarrow K_2$

Table R

Attack idea:

- Build L,R tables
- Find i, j s.t. $L[i] = R[j]$
- Guess that $K_1K_2 = T_iT_j$

Meet-in-the-middle attack on 2DES

Let $T_1, \ldots, T_{2^{56}}$ denote an enumeration of DES keys.

```
adversary $A_{\rm MinM}$
$M_1 \leftarrow 0^{64}$; $C_1 \leftarrow \mathbf{Fn}(M_1)$
for $i = 1, \ldots, 2^{56}$ do $L[i] \leftarrow \mathrm{DES}(T_i, M_1)$
for $j = 1, \ldots, 2^{56}$ do $R[j] \leftarrow \mathrm{DES}^{-1}(T_j, C_1)$
$S \leftarrow \{ (i, j) : L[i] = R[j] \}$
Pick some $(l, r) \in S$ and return $T_l \,\|\, T_r$
```

Attack takes about 2^{57} DES/DES^{-1} computations and has $\mathbf{Adv}^{\rm kr}_{\rm 2DES}(A_{\rm MinM}) = 1$.

This uses $q = 1$ and is unlikely to return the target key. For that one should extend the attack to a larger value of q.

3DES

Block ciphers

$3DES3 : \{0,1\}^{168} \times \{0,1\}^{64} \to \{0,1\}^{64}$

$3DES2 : \{0,1\}^{112} \times \{0,1\}^{64} \to \{0,1\}^{64}$

are defined by

$$3DES3_{K_1 \,\|\, K_2 \,\|\, K_3}(M) \;=\; DES_{K_3}(DES^{-1}_{K_2}(DES_{K_1}(M)))$$

$$3DES2_{K_1 \,\|\, K_2}(M) \;=\; DES_{K_2}(DES^{-1}_{K_1}(DES_{K_2}(M)))$$

Meet-in-the-middle attack on 3DES3 reduces its "effective" key length to 112.

Block size limitation

Later we will see "birthday" attacks that "break" a block cipher $E : \{0,1\}^k \times \{0,1\}^\ell \to \{0,1\}^\ell$ in time $2^{\ell/2}$

For DES this is $2^{64/2} = 2^{32}$ which is small, and this is unchanged for 2DES and 3DES.

Would like a larger block size.

AES

1998: NIST announces competition for a new block cipher

- key length 128
- block length 128
- faster than DES in software

Submissions from all over the world: MARS, Rijndael, Two-Fish, RC6, Serpent, Loki97, Cast-256, Frog, DFC, Magenta, E2, Crypton, HPC, Safer+, Deal

AES

1998: NIST announces competition for a new block cipher

- key length 128
- block length 128
- faster than DES in software

Submissions from all over the world: MARS, Rijndael, Two-Fish, RC6, Serpent, Loki97, Cast-256, Frog, DFC, Magenta, E2, Crypton, HPC, Safer+, Deal

2001: NIST selects Rijndael to be AES.

AES

```
function $\mathrm{AES}_K(M)$
    $(K_0, \ldots, K_{10}) \leftarrow \mathit{expand}(K)$
    $s \leftarrow M \oplus K_0$
    for $r = 1$ to $10$ do
        $s \leftarrow S(s)$
        $s \leftarrow \mathit{shift\text{-}rows}(s)$
        if $r \leq 9$ then $s \leftarrow \mathit{mix\text{-}cols}(s)$ fi
        $s \leftarrow s \oplus K_r$
    end for
    return $s$
```

- Fewer tables than DES
- Finite field operations

The AES movie

`http://www.youtube.com/watch?v=H2LlHOw_ANg`

Implementing AES

	Code size	**Performance**
Pre-compute and store round function tables	largest	fastest
Pre-compute and store S-boxes only	smaller	slower
No pre-computation	smallest	slowest

AES-NI: Hardware for AES, now present on most processors. Your laptop may have it! Can run AES at around 1 cycle/byte. VERY fast!

Security of AES

Best known key-recovery attack [BoKhRe11] takes $2^{126.1}$ time, which is only marginally better than the 2^{128} time of EKS.

There are attacks on reduced-round versions of AES as well as on its sibling algorithms AES192, AES256. Many of these are "related-key" attacks. There are also effective side-channel attacks on AES such as "cache-timing" attacks [Be05,OsShTr05].

Exercise

Define F: $\{0,1\}^{256} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$ by

```
**Alg** $F_{K_1\|K_2}(x_1\|x_2)$
$y_1 \leftarrow \text{AES}^{-1}(K_1, x_1 \oplus x_2)$; $y_2 \leftarrow \text{AES}(K_2, \overline{x_2})$
Return $y_1\|y_2$
```

for all 128-bit strings K_1, K_2, x_1, x_2, where $\overline{x}$ denotes the bitwise complement of x. (For example $\overline{01} = 10$.) Let T_{AES} denote the time for one computation of AES or AES^{-1}. Below, running times are worst-case and should be functions of T_{AES}.

Exercise

1. Prove that F is a blockcipher.
2. What is the running time of a 4-query exhaustive key-search attack on F?
3. Give a 4-query key-recovery attack in the form of an adversary A specified in pseudocode, achieving $\mathbf{Adv}^{\mathrm{kr}}_F(A) = 1$ and having running time $\mathcal{O}(2^{128} \cdot T_{\text{AES}})$ where the big-oh hides some small constant.

Limitations of security against key recovery

So far, a block cipher has been viewed as secure if it resists key recovery, meaning there is no efficient adversary A having $\mathbf{Adv}^{\mathrm{kr}}_E(A) \approx 1$.

Is security against key recovery enough?

Not really. For example define E: $\{0,1\}^{128} \times \{0,1\}^{256} \rightarrow \{0,1\}^{256}$ by

$$E_K(M[1]M[2]) = M[1]\|\text{AES}_K(M[2])$$

This is as secure against key-recovery as AES, but not a "good" blockcipher because half the message is in the clear in the ciphertext.

So what?

Possible reaction: But DES, AES are not designed like E above, so why does this matter?

Answer: It tells us that security against key recovery is not, as a block-cipher property, sufficient for security of uses of the block cipher.

As designers and users we want to know what properties of a block cipher give us security when the block cipher is used.

So what is a “good” block cipher?

Possible Properties	Necessary?	Sufficient?
security against key recovery	YES	NO!
hard to find M given $C = E_K(M)$	YES	NO!
$\vdots$		

We can't define or understand security well via some such (indeterminable) list.

We want a single “master” property of a block cipher that is sufficient to ensure security of common usages of the block cipher.