BLOCK CIPHERS
and
KEY-RECOVERY SECURITY
There are only 10 types of people in the world: Those who understand binary and those who don't.
\{0, 1\}^n is the set of \(n\)-bit strings and \(\{0, 1\}^*\) is the set of all strings of finite length. By \(\varepsilon\) 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: \(|0100| = 4\).

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

By \(x \leftarrow S\) we denote picking an element at random from set \(S\) and assigning it to \(x\). Thus \(\Pr[x = s] = 1/|S|\) for every \(s \in S\).
Let $n \geq 1$ be an integer. Let $X_1, \ldots, X_n$ and $Y$ be (non-empty) sets.

By $f : X_1 \times \cdots \times X_n \to Y$ 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 $y = f(x_1, \ldots, x_n) \in Y$.

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 $f$.

**Example:** Define $f : \mathbb{Z}_2 \times \mathbb{Z}_3 \to \mathbb{Z}_3$ by $f(x_1, x_2) = (x_1 + x_2) \mod 3$. This is a function with $n = 2$ inputs, domain $\mathbb{Z}_2 \times \mathbb{Z}_3$ and range $\mathbb{Z}_3$. 
Suppose $f : X \to Y$ 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 $f^{-1} : Y \to X$ such that $f^{-1}(f(x)) = x$ for all $x \in X$.

This means $f$ must be one-to-one and onto: for every $y \in Y$ there is a unique $x \in X$ such that $f(x) = y$. 
Consider the following two functions $f : \{0, 1\}^2 \rightarrow \{0, 1\}^2$, where $X = Y = \{0, 1\}^2$:

<table>
<thead>
<tr>
<th>$x$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>01</td>
<td>11</td>
<td>00</td>
<td>10</td>
</tr>
</tbody>
</table>

A permutation

<table>
<thead>
<tr>
<th>$x$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>01</td>
<td>11</td>
<td>11</td>
<td>10</td>
</tr>
</tbody>
</table>

Not a permutation
Consider the following two functions $f : \{0, 1\}^2 \rightarrow \{0, 1\}^2$, where $X = Y = \{0, 1\}^2$:

<table>
<thead>
<tr>
<th>$x$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>01</td>
<td>11</td>
<td>00</td>
<td>10</td>
</tr>
</tbody>
</table>

A permutation

<table>
<thead>
<tr>
<th>$x$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>01</td>
<td>11</td>
<td>11</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

Not a permutation

<table>
<thead>
<tr>
<th>$x$</th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f^{-1}(x)$</td>
<td>10</td>
<td>00</td>
<td>11</td>
<td>01</td>
</tr>
</tbody>
</table>

Its inverse
A family of functions (also called a function family) is a two-input function $F : \text{Keys} \times D \to R$. For $K \in \text{Keys}$ we let $F_K : D \to R$ be defined by $F_K(x) = F(K, x)$ for all $x \in D$.

- 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 $\ell$ 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 $F : \mathbb{Z}_2 \times \mathbb{Z}_3 \to \mathbb{Z}_3$ by $F(K, x) = (K \cdot x) \mod 3$.
- This is a family of functions with domain $\mathbb{Z}_2 \times \mathbb{Z}_3$ and range $\mathbb{Z}_3$.
- If $K = 1$ then $F_K : \mathbb{Z}_3 \to \mathbb{Z}_3$ is given by $F_K(x) = x \mod 3$. 
Let $E: \text{Keys} \times D \to R$ be a family of functions. We say that $E$ is a **block cipher** if

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

We let $E^{-1} : \text{Keys} \times D \to D$, defined by $E^{-1}(K, y) = E^{-1}_K(y)$, 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 $D = \{0, 1\}^\ell$ we call $\ell$ the block length.
Block cipher $E$: $\{0, 1\}^2 \times \{0, 1\}^2 \rightarrow \{0, 1\}^2$ (left), where the table entry corresponding to the key in row $K$ and input in column $x$ is $E_K(x)$. Its inverse $E^{-1}$: $\{0, 1\}^2 \times \{0, 1\}^2 \rightarrow \{0, 1\}^2$ (right).

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>11</td>
<td>00</td>
<td>10</td>
<td>01</td>
</tr>
<tr>
<td>01</td>
<td>11</td>
<td>10</td>
<td>01</td>
<td>00</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>11</td>
<td>00</td>
<td>01</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>00</td>
<td>10</td>
<td>01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>01</td>
<td>11</td>
<td>10</td>
<td>00</td>
</tr>
<tr>
<td>01</td>
<td>11</td>
<td>10</td>
<td>01</td>
<td>00</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>11</td>
<td>00</td>
<td>01</td>
</tr>
<tr>
<td>11</td>
<td>01</td>
<td>11</td>
<td>10</td>
<td>00</td>
</tr>
</tbody>
</table>

- 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
Let $\ell = k$ and define $E : \{0, 1\}^k \times \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ 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

$$E^{-1}(K, y) = E_K^{-1}(y) = K \oplus y$$
Let $E \colon \text{Keys} \times D \to D$ 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.
Slow is good

THINKING, FAST AND SLOW

DANIEL KAHNEMAN
Let $E : \text{Keys} \times D \to D$ 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.
Above we had given the following example of a family of functions: 

\[ F : \mathbb{Z}_2 \times \mathbb{Z}_3 \to \mathbb{Z}_3 \] 

defined by \( F(K, x) = (K \cdot x) \mod 3 \).

**Question:** Is \( F \) a block cipher? Why or why not?
Above we had given the following example of a family of functions: 
\[ F : \mathbb{Z}_2 \times \mathbb{Z}_3 \rightarrow \mathbb{Z}_3 \] 
defined by 
\[ F(K, x) = (K \cdot x) \mod 3. \]

**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.
Above we had given the following example of a family of functions:

\[ F: \mathbb{Z}_2 \times \mathbb{Z}_3 \rightarrow \mathbb{Z}_3 \text{ defined by } F(K, x) = (K \cdot x) \mod 3. \]

**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?
Above we had given the following example of a family of functions: \( F : \mathbb{Z}_2 \times \mathbb{Z}_3 \rightarrow \mathbb{Z}_3 \) defined by \( F(K, x) = (K \cdot x) \mod 3 \).

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

- $K \leftarrow \{0, 1\}^k$ is known to parties $S$, $R$, but not given to adversary $A$.
- $S$, $R$ 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 $y_i$; ...
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 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

Key Length $k = 56$

Block length $\ell = 64$

So,

$$\text{DES}: \{0, 1\}^{56} \times \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}$$

$$\text{DES}^{-1}: \{0, 1\}^{56} \times \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}$$
function \( \text{DES}_K(M) \)  
\[
\begin{align*}
(K_1, \ldots, K_{16}) & \leftarrow \text{KeySchedule}(K) \quad // |K_i| = 48 \text{ for } 1 \leq i \leq 16 \\
M & \leftarrow \text{IP}(M) \\
\text{Parse } M \text{ as } L_0 \parallel R_0 \\
\text{for } i = 1 \text{ to } 16 \text{ do} \\
L_i & \leftarrow R_{i-1} \\
R_i & \leftarrow f(K_i, R_{i-1}) \oplus L_{i-1} \\
C & \leftarrow \text{IP}^{-1}(L_{16} \parallel R_{16}) \\
\text{return } C
\end{align*}
\]

**Round i:**

**Invertible given \( K_i \):**
function DES_K(M)  
// |K| = 56 and |M| = 64
(K_1, \ldots, K_{16}) \leftarrow \text{KeySchedule}(K)  
// |K_i| = 48 for 1 \leq i \leq 16
M \leftarrow \text{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 \text{IP}^{-1}(L_{16} \| R_{16})
return C

function DES^{-1}_K(C)  
// |K| = 56 and |M| = 64
(K_1, \ldots, K_{16}) \leftarrow \text{KeySchedule}(K)  
// |K_i| = 48 for 1 \leq i \leq 16
C \leftarrow \text{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 \text{IP}^{-1}(L_0 \| R_0)
return M
function \texttt{DES}_K(M) \quad // \quad |K| = 56 \text{ and } |M| = 64

(K_1, \ldots, K_{16}) \leftarrow \text{KeySchedule}(K) \quad // \quad |K_i| = 48 \text{ for } 1 \leq i \leq 16

M \leftarrow \text{IP}(M)

Parse $M$ as $L_0 \parallel R_0$ \quad // \quad |L_0| = |R_0| = 32

for $i = 1$ to $16$ do

\quad $L_i \leftarrow R_{i-1}$

\quad $R_i \leftarrow f(K_i, R_{i-1}) \oplus L_{i-1}$

end for

$C \leftarrow \text{IP}^{-1}(L_{16} \parallel R_{16})$

return $C$

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

\[
\begin{array}{cccccccccc}
40 & 8 & 48 & 16 & 56 & 24 & 64 & 32 \\
39 & 7 & 47 & 15 & 55 & 23 & 63 & 31 \\
38 & 6 & 46 & 14 & 54 & 22 & 62 & 30 \\
37 & 5 & 45 & 13 & 53 & 21 & 61 & 29 \\
36 & 4 & 44 & 12 & 52 & 20 & 60 & 28 \\
35 & 3 & 43 & 11 & 51 & 19 & 59 & 27 \\
34 & 2 & 42 & 10 & 50 & 18 & 58 & 26 \\
33 & 1 & 41 & 9 & 49 & 17 & 57 & 25 \\
\end{array}
\]
function $f(J, R)$  // $|J| = 48$ and $|R| = 32$

$R \leftarrow E(R) ; \quad R \leftarrow R \oplus J$

Parse $R$ as $R_1 || R_2 || R_3 || R_4 || R_5 || R_6 || R_7 || R_8$ // $|R_i| = 6$ for $1 \leq i \leq 8$

for $i = 1, \ldots, 8$ do

\[ R_i \leftarrow S_i(R_i) \]  // 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

$R \leftarrow P(R) ; \text{return } R$

\[
\begin{array}{cccccccc}
32 & 1 & 2 & 3 & 4 & 5 & 16 & 7 & 20 & 21 \\
4 & 5 & 6 & 7 & 8 & 9 & 29 & 12 & 28 & 17 \\
8 & 9 & 10 & 11 & 12 & 13 & 1 & 15 & 23 & 26 \\
12 & 13 & 14 & 15 & 16 & 17 & 5 & 18 & 31 & 10 \\
16 & 17 & 18 & 19 & 20 & 21 & 2 & 8 & 24 & 14 \\
20 & 21 & 22 & 23 & 24 & 25 & 32 & 27 & 3 & 9 \\
24 & 25 & 26 & 27 & 28 & 29 & 19 & 13 & 30 & 6 \\
28 & 29 & 30 & 31 & 32 & 1 & 22 & 11 & 4 & 25 \\
\end{array}
\]
### S-boxes

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S₁</strong></td>
<td>0</td>
<td>14</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>11</td>
<td>8</td>
<td>3</td>
<td>10</td>
<td>6</td>
<td>12</td>
<td>5</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>15</td>
<td>7</td>
<td>4</td>
<td>14</td>
<td>2</td>
<td>13</td>
<td>1</td>
<td>10</td>
<td>6</td>
<td>12</td>
<td>11</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>14</td>
<td>8</td>
<td>13</td>
<td>6</td>
<td>2</td>
<td>11</td>
<td>15</td>
<td>12</td>
<td>9</td>
<td>7</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>15</td>
<td>12</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>9</td>
<td>1</td>
<td>7</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>14</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S₂</strong></td>
<td>0</td>
<td>15</td>
<td>1</td>
<td>8</td>
<td>14</td>
<td>6</td>
<td>11</td>
<td>3</td>
<td>4</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>13</td>
<td>12</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>13</td>
<td>4</td>
<td>7</td>
<td>15</td>
<td>2</td>
<td>8</td>
<td>14</td>
<td>12</td>
<td>0</td>
<td>1</td>
<td>10</td>
<td>6</td>
<td>9</td>
<td>11</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>14</td>
<td>7</td>
<td>11</td>
<td>10</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>5</td>
<td>8</td>
<td>12</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>1</td>
<td>3</td>
<td>15</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S₃</strong></td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>9</td>
<td>14</td>
<td>6</td>
<td>3</td>
<td>15</td>
<td>5</td>
<td>1</td>
<td>13</td>
<td>12</td>
<td>7</td>
<td>11</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>13</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>14</td>
<td>12</td>
<td>11</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>13</td>
<td>6</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>15</td>
<td>3</td>
<td>0</td>
<td>11</td>
<td>1</td>
<td>2</td>
<td>12</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>13</td>
<td>0</td>
<td>6</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>15</td>
<td>14</td>
<td>3</td>
<td>11</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>
Let $E: \text{Keys} \times \text{D} \rightarrow \text{R}$ be a block cipher known to the adversary $A$.

- Sender Alice and receiver Bob share a target key $K \in \text{Keys}$.
- Alice encrypts $M_i$ to get $C_i = E_K(M_i)$ for $1 \leq i \leq q$, 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

<table>
<thead>
<tr>
<th>Game $\text{TKR}_E$</th>
<th>procedure $\text{Fn}(M)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>procedure Initialize</td>
<td>Return $E(K, M)$</td>
</tr>
<tr>
<td>$K \leftarrow \text{Keys}$</td>
<td>procedure $\text{Finalize}(K')$</td>
</tr>
<tr>
<td></td>
<td>Return $(K = K')$</td>
</tr>
</tbody>
</table>

**Definition:** $\text{Adv}_{E}^{\text{tkr}}(A) = \Pr[\text{TKR}_E^A \Rightarrow \text{true}]$.

- First **Initialize** executes, selecting *target key* $K \leftarrow \text{Keys}$, but not giving it to $A$.
- Now $A$ can call (query) **Fn** on any input $M \in D$ 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 $E: \text{Keys} \times D \rightarrow R$ be a family of functions. We say that key $K' \in \text{Keys}$ 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

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>11</td>
<td>00</td>
<td>10</td>
<td>01</td>
</tr>
<tr>
<td>01</td>
<td>11</td>
<td>10</td>
<td>01</td>
<td>00</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>11</td>
<td>00</td>
<td>01</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>00</td>
<td>10</td>
<td>01</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Game $\text{KR}_E$</th>
<th>procedure Initialize</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K \leftarrow \text{Keys}; i \leftarrow 0$</td>
<td>$K' \leftarrow 0$</td>
</tr>
<tr>
<td>procedure $\text{Fn}(M)$</td>
<td>procedure $\text{Finalize}(K')$</td>
</tr>
<tr>
<td>$i \leftarrow i + 1; M_i \leftarrow M$</td>
<td>win $\leftarrow$ true</td>
</tr>
<tr>
<td>$C_i \leftarrow E(K, M_i)$</td>
<td>For $j = 1, \ldots, i$ do</td>
</tr>
<tr>
<td>Return $C_i$</td>
<td>If $E(K', M_j) \neq C_j$ then win $\leftarrow$ false</td>
</tr>
<tr>
<td></td>
<td>If $M_j \in {M_1, \ldots, M_{j-1}}$ then win $\leftarrow$ false</td>
</tr>
<tr>
<td></td>
<td>Return win</td>
</tr>
</tbody>
</table>

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

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 $\text{Fn}$ oracle.
kr advantage always exceeds tkr advantage

Fact: Suppose that, in game KRₐ, adversary A makes queries M₁, . . . , Mₙ to Fn, thereby defining C₁, . . . , Cₙ. Then the target key K is consistent with (M₁, C₁), . . . , (Mₙ, Cₙ).

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

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

Why? If the Kᵢ that A returns equals the target key K, then, by the Fact, the input-output examples (M₁, C₁), . . . , (Mₙ, Cₙ) will of course be consistent with Kᵢ.
Exhaustive Key Search attack

Let $E: \text{Keys} \times D \rightarrow R$ be a function family with $\text{Keys} = \{ T_1, \ldots, T_N \}$ 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 $M_j \leftarrow x_j$; $C_j \leftarrow F_n(M_j)$

For $i = 1, \ldots, N$ do

$\quad$ if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$

**Question:** What is $\text{Adv}_E^{kr}(A_{eks})$?
Exhaustive Key Search attack

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

**adversary $A_{\text{eks}}$**

For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow F_n(M_j)$

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 $\text{Adv}^k_E(A_{\text{eks}})$?

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

**adversary $A_{\text{eks}}$**

For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow F_n(M_j)$

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 $\text{Adv}_E^{\text{tkr}}(A_{\text{eks}})$?
Exhaustive Key Search attack

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

**adversary** \( A_{\text{eks}} \)

For \( j = 1, \ldots, q \) do \( M_j \leftarrow x_j; \ C_j \leftarrow F_n(M_j) \)

For \( i = 1, \ldots, N \) do

\[
\text{if } (\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j) \text{ then return } T_i
\]

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

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

In practice if \( E : \{0, 1\}^k \times \{0, 1\}^\ell \to \{0, 1\}^\ell \) is a “real” block cipher and \( q > k/\ell \), we expect that \( \text{Adv}^{tkr}_E(A_{\text{eks}}) \) is close to 1 because \( K \) is likely the only key consistent with the input-output examples.
Let $k, \ell \geq 1$ be given integers. Present in pseudocode a block cipher $E: \{0, 1\}^k \times \{0, 1\}^\ell \rightarrow \{0, 1\}^\ell$ for which you do the following:

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

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

Exercise: tkr advantage can be much less than kr
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 \(A_{\text{eks}} (q = 1)\) to succeed in \(2^{55}\) DES computations, so it takes time

\[
\frac{2^{55}}{2.5 \times 10^7} \approx 1.4 \times 10^9 \text{ seconds}
\]

\[
\approx 45 \text{ years!}
\]

Key Complementation \(\Rightarrow\) 22.5 years

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

<table>
<thead>
<tr>
<th>Attack</th>
<th>when</th>
<th>$q$, running time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Differential cryptanalysis</td>
<td>1992</td>
<td>$2^{47}$</td>
</tr>
<tr>
<td>Linear cryptanalysis</td>
<td>1993</td>
<td>$2^{44}$</td>
</tr>
</tbody>
</table>
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:

<table>
<thead>
<tr>
<th>Attack</th>
<th>when</th>
<th>$q$, running time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Differential cryptanalysis</td>
<td>1992</td>
<td>$2^{47}$</td>
</tr>
<tr>
<td>Linear cryptanalysis</td>
<td>1993</td>
<td>$2^{44}$</td>
</tr>
</tbody>
</table>

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

In practice these attacks were prohibitively expensive.
**EKS revisited**

**adversary** $A_{eks}$

For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow F_n(M_j)$

For $i = 1, \ldots, N$ do

if $(\forall j \in \{1, \ldots, q\} : E(T_i, M_j) = C_j)$ then return $T_i$
adversary $A_{eks}$

For $j = 1, \ldots, q$ do $M_j \leftarrow x_j; \ C_j \leftarrow F_n(M_j)$

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 $A_{eks}$

For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow F_n(M_j)$

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

\[ K \leftarrow_{\$} \{0, 1\}^{56} ; \ Y \leftarrow \text{DES}(K, X) ; \text{Publish} \ Y \text{ on website.} \]

Reward for recovering \( X \)

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Post Date</th>
<th>Reward</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1997</td>
<td>$10,000</td>
<td>Distributed.Net: 4 months</td>
</tr>
<tr>
<td>II</td>
<td>1998</td>
<td>Depends how fast you find key</td>
<td>Distributed.Net: 41 days. EFF: 56 hours</td>
</tr>
<tr>
<td>III</td>
<td>1998</td>
<td>As above</td>
<td>&lt; 28 hours</td>
</tr>
</tbody>
</table>
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.
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 $K_1 K_2$ is a target 2DES key and adversary has $M, C$ such that

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

Then

$$DES^{-1}_{K_2}(C) = DES_{K_1}(M)$$
Meet-in-the-middle attack on 2DES

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

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$DES(T_1, M)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_i$</td>
<td>$DES(T_i, M)$</td>
</tr>
<tr>
<td>$T_N$</td>
<td>$DES(T_N, M)$</td>
</tr>
</tbody>
</table>

Table $L$

<table>
<thead>
<tr>
<th>$DES^{-1}(T_1, C)$</th>
<th>$T_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$DES^{-1}(T_j, C)$</td>
<td>$T_j$</td>
</tr>
<tr>
<td>$DES^{-1}(T_N, C)$</td>
<td>$T_N$</td>
</tr>
</tbody>
</table>

Table $R$

Attack idea:
- Build L,R tables

Mihir Bellare  
UCSD
Meet-in-the-middle attack on 2DES

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

<table>
<thead>
<tr>
<th>$K_1 \rightarrow$</th>
<th>$T_1$</th>
<th>$DES(T_1, M)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_i$</td>
<td>$DES(T_i, M)$</td>
<td></td>
</tr>
<tr>
<td>$T_N$</td>
<td>$DES(T_N, M)$</td>
<td>$Table \ L$</td>
</tr>
</tbody>
</table>

$DES_{K_2}^{-1}(T_1, C)$ $T_1$

$DES_{K_2}^{-1}(T_j, C)$ $T_j$

$DES_{K_2}^{-1}(T_N, C)$ $T_N$

$\leftarrow K_2$

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** $A_{\text{MinM}}$

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

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

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

3DES3 : \{0, 1\}^{168} \times \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}
3DES2 : \{0, 1\}^{112} \times \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}

are defined by

\[
3DES_{K_1 \parallel K_2 \parallel K_3}(M) = \text{DES}_{K_3}(\text{DES}^{-1}_{K_2}(\text{DES}_{K_1}(M)))
\]

\[
3DES_{K_1 \parallel K_2}(M) = \text{DES}_{K_2}(\text{DES}^{-1}_{K_1}(\text{DES}_{K_2}(M)))
\]

Meet-in-the-middle attack on 3DES3 reduces its “effective” key length to 112.
Later we will see “birthday” attacks that “break” a block cipher
\[ E : \{0,1\}^k \times \{0,1\}^\ell \rightarrow \{0,1\}^\ell \] 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.
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
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.
function $\text{AES}_K(M)$

$(K_0, \ldots, K_{10}) \leftarrow \text{expand}(K)$

$s \leftarrow M \oplus K_0$

for $r = 1$ to $10$ do

\begin{center}
\begin{itemize}
  \item $s \leftarrow S(s)$
  \item $s \leftarrow \text{shift-rows}(s)$
  \item if $r \leq 9$ then $s \leftarrow \text{mix-cols}(s)$ fi
  \item $s \leftarrow s \oplus K_r$
\end{itemize}
\end{center}

end for

return $s$

• Fewer tables than DES
• Finite field operations
The AES movie

http://www.youtube.com/watch?v=H2L1HOw_ANg
<table>
<thead>
<tr>
<th>Pre-compute and store round function tables</th>
<th>Pre-compute and store S-boxes only</th>
<th>No pre-computation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code size: largest</td>
<td>Performance: fastest</td>
<td>Performance: smallest</td>
</tr>
<tr>
<td>Pre-compute and store S-boxes only</td>
<td>Performance: slower</td>
<td>Performance: slowest</td>
</tr>
</tbody>
</table>

**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!
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

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

for all 128-bit strings $K_1, K_2, x_1, x_2$, where $\overline{x}$ denotes the bitwise complement of $x$. (For example $01 = 10$.) Let $T_{\text{AES}}$ denote the time for one computation of AES or AES$^{-1}$. Below, running times are worst-case and should be functions of $T_{\text{AES}}$. 
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 $\text{Adv}_F^{kr}(A) = 1$ and having running time $O(2^{128} \cdot T_{\text{AES}})$ 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 $\text{Adv}_E^{kr}(A) \approx 1$.

Is security against key recovery enough?

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


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.
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?

<table>
<thead>
<tr>
<th>Possible Properties</th>
<th>Necessary?</th>
<th>Sufficient?</th>
</tr>
</thead>
<tbody>
<tr>
<td>security against key recovery</td>
<td>YES</td>
<td>NO!</td>
</tr>
<tr>
<td>hard to find $M$ given $C = E_K(M)$</td>
<td>YES</td>
<td>NO!</td>
</tr>
</tbody>
</table>

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.