Lecture 2 – Blockciphers and key recovery security

CS-466 Applied Cryptography

Adam O’Neill

Setting the Stage

Perfect security => keys as long as messages.

Setting the Stage

Perfect security => keys as long as messages.

From now on we move to the setting of computationally-bounded adversaries.

Setting the Stage

Perfect security => keys as long as messages.

From now on we move to the setting of computationally-bounded adversaries.

Today: first lower-level primitive, blockciphers

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: $|0100| = 4$.

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

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

Functions

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 $y = f(x_1, \ldots, x_n) \in Y$.

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 $f\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ 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

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 $x \in X$.

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

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

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

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

Example: Define $F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ 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 $F_K(x) = x \bmod 3$.

What is a blockcipher?

Let E: Keys $\times$ D $\rightarrow$ 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 $\rightarrow$ D is a permutation for every key $K \in$ Keys, meaning has an inverse E_K^{-1}: D $\rightarrow$ D such that $E_K^{-1}(E_K(x)) = x$ for all $x \in$ D.

We let E^{-1}: Keys $\times$ D $\rightarrow$ 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 Keys $= \{0,1\}^k$ then k is the key length as before. If D $= \{0,1\}^\ell$ we call ℓ the block length.

Blockcipher Examples

Block cipher E: $\{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ (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}: $\{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ (right).

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

	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

Other examples?

$E_k(x) = k \oplus x$ (OTP)

$E_k(x) = x$ (identity)

Exercise

Let E: Keys $\times$ D $\rightarrow$ D be a block cipher. Is E a permutation?

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

permutation doesn't make sense for two-argument function

Another Exercise

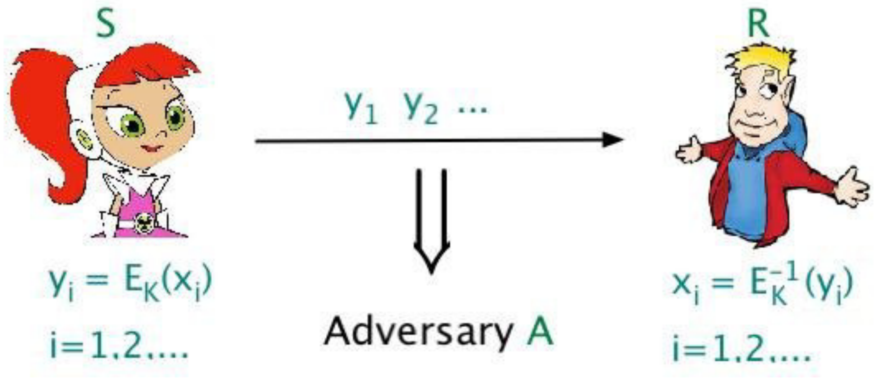
Above we had given the following example of a family of functions: $F\colon \mathbf{Z}_2 \times \mathbf{Z}_3 \to \mathbf{Z}_3$ defined by $F(K, x) = (K \cdot x) \bmod 3$.

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

Blockcipher Usage

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

- $K \overset{\$}{\leftarrow} \{0,1\}^k$ is known to parties S, R, but not given to adversary A.
- S, R use E_K for encryption



S

$y_1 \; y_2 \; ...$

R

$y_i = E_K(x_i)$

$i=1,2,...$

Adversary A

$x_i = E_K^{-1}(y_i)$

$i=1,2,...$

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

Shannon’s Design Criterion (Informal)

Shannon’s Design Criterion (Informal)

- Confusion: Each bit of the output should depend on many bits of the input

Shannon’s Design Criterion (Informal)

- Confusion: Each bit of the output should depend on many bits of the input
- Diffusion: Changing one bit of the input should “re-randomize” the entire output (avalanche effect)

Shannon’s Design Criterion (Informal)

- Confusion: Each bit of the output should depend on many bits of the input
- Diffusion: Changing one bit of the input should “re-randomize” the entire output (avalanche effect)
- Not really solved (for many input-outputs) until much later: Data Encryption Standard (DES)

History of DES

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.

DES Parameters

Key Length $k = 56$

Block length $\ell = 64$

So,

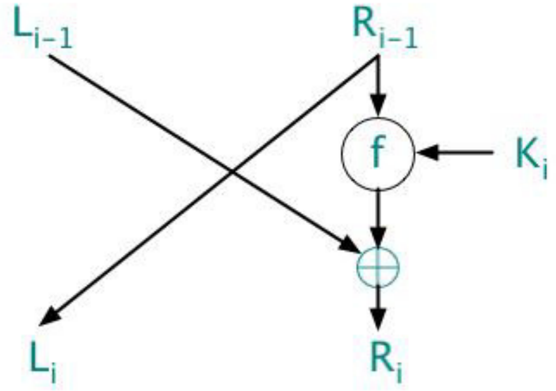
$$\mathsf{DES}\colon \{0,1\}^{56} \times \{0,1\}^{64} \to \{0,1\}^{64}$$

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

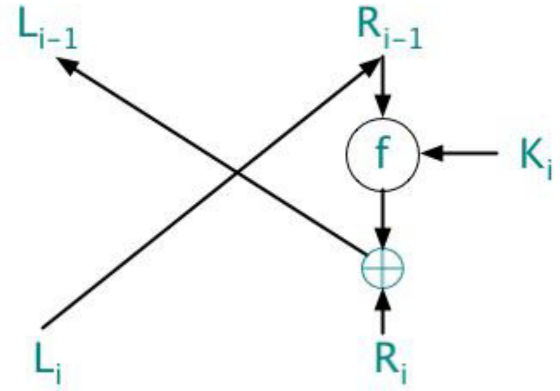
DES Construction

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

Round i:



Invertible given K_i:



Inverse

```
function $\mathsf{DES}_K(M)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \leq i \leq 16$
    $M \leftarrow IP(M)$
    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 $\mathsf{DES}_K^{-1}(C)$    // $|K| = 56$ and $|M| = 64$
    $(K_1, \ldots, K_{16}) \leftarrow KeySchedule(K)$    // $|K_i| = 48$ for $1 \leq i \leq 16$
    $C \leftarrow IP(C)$
    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 $M$
```

Round function

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

Key-Recovery Attacks

Let E: Keys × D → 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 $1 \leq i \leq q$, 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!

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 Game

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

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

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

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

- First **Initialize** executes, selecting *target key* $K \xleftarrow{\$} \mathsf{Keys}$, but not giving it to A.
- Now A can call (query) **Fn** on any input $M \in \mathsf{D}$ 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 × D → 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: $\{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ 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

Let E: Keys × D → R be a family of functions, and A an adversary.

```
Game $\mathrm{KR}_E$

procedure Initialize
$K \xleftarrow{\$}$ 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: $\mathbf{Adv}^{\mathrm{kr}}_{E}(A) = \Pr[\mathrm{KR}^{A}_{E} \Rightarrow \text{true}]$.

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 **Fn** oracle.

A relation

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

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

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

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

```
adversary $A_{\text{eks}}$
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
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 $\mathbf{Adv}_E^{\text{kr}}(A_{\text{eks}})$?

1

Exhaustive Key Search

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

```
adversary $A_{\text{eks}}$
For $j = 1, \ldots, q$ do $M_j \leftarrow x_j$; $C_j \leftarrow \mathbf{Fn}(M_j)$
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 $\mathbf{Adv}^{\text{tkr}}_{E}(A_{\text{eks}})$?

?

Exhaustive Key Search

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

```
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 $T_i$
```

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.

Exhaustive Key-Search on DES

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 $\Rightarrow$ 22.5 years

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

generic attack

Differential & 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}

non generic attack

An observation

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 Summary

$K \xleftarrow{\$} \{0,1\}^{56}$; $Y \leftarrow \mathrm{DES}(K, X)$; 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

Increasing Key-Length

Can one use DES to design a new blockcipher with longer effective key-length?

2DES

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

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

2DES

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

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

- 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

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

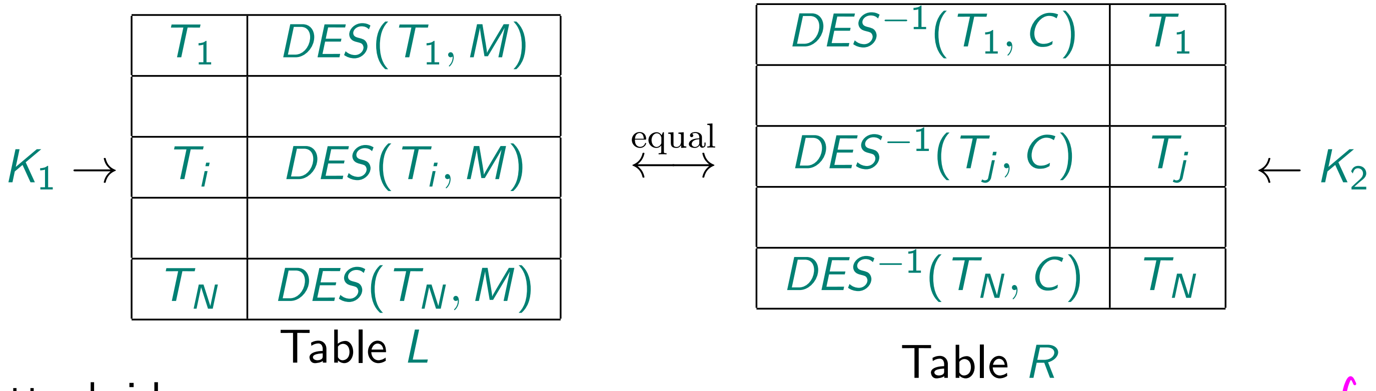
$$C = 2DES_{K_1K_2}(M) = DES_{K_2}(DES_{K_1}(M))$$

Then

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

Meet-in-the-Middle Attack

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

112 : physical key length

57 : effective key length

Attack idea:

- Build L,R tables
- Find i, j s.t. $L[i] = R[j]$
- Guess that $K_1 K_2 = T_i T_j$

4 query EKS : $2^{112} \cdot 8\, T_{DES} + 4$ Fn queries

Best attack: $2^{57} \cdot 8\, T_{DES} + 4$ Fn queries

Translating to Pseudocode

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

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

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

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

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

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

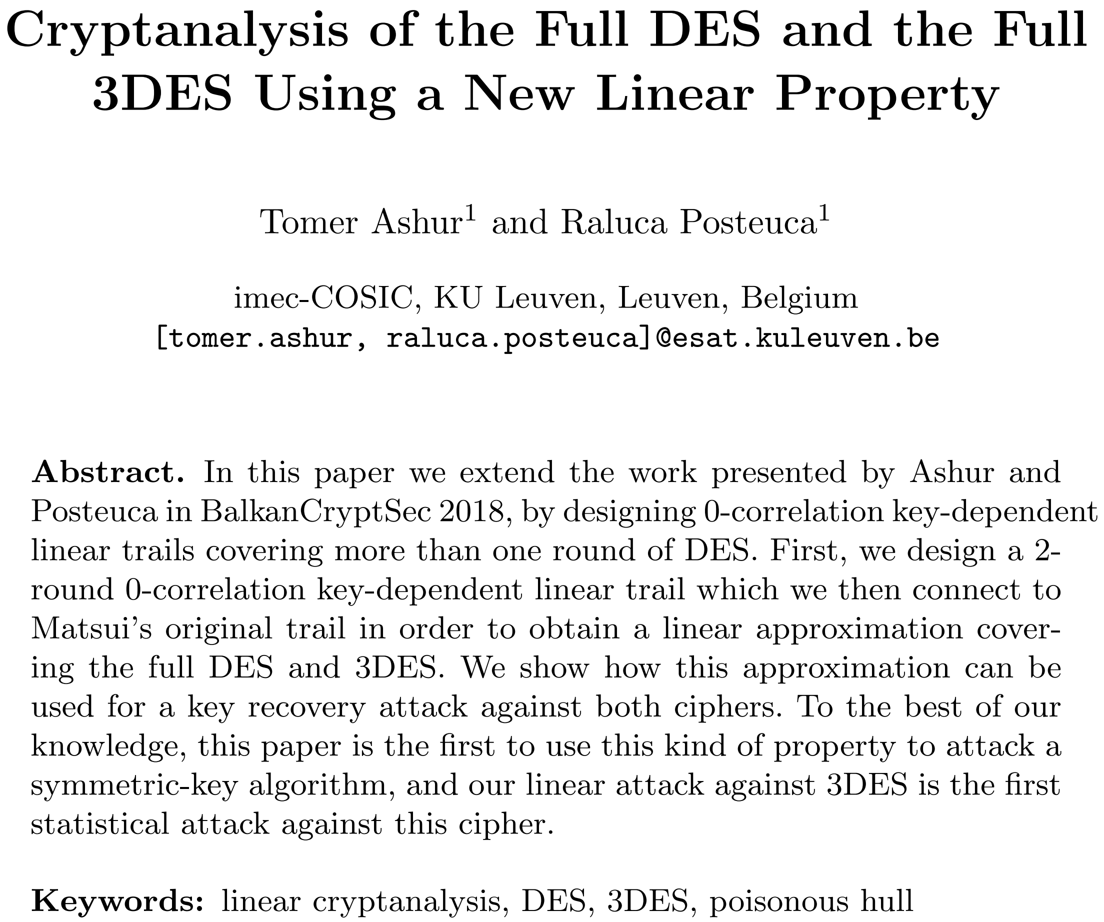
are defined by

$$\text{3DES3}_{K_1 \,\|\, K_2 \,\|\, K_3}(M) \;=\; \text{DES}_{K_3}(\text{DES}^{-1}_{K_2}(\text{DES}_{K_1}(M))$$

$$\text{3DES2}_{K_1 \,\|\, K_2}(M) \;=\; \text{DES}_{K_2}(\text{DES}^{-1}_{K_1}(\text{DES}_{K_2}(M))$$

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

Better Attacks?



Cryptanalysis of the Full DES and the Full 3DES Using a New Linear Property

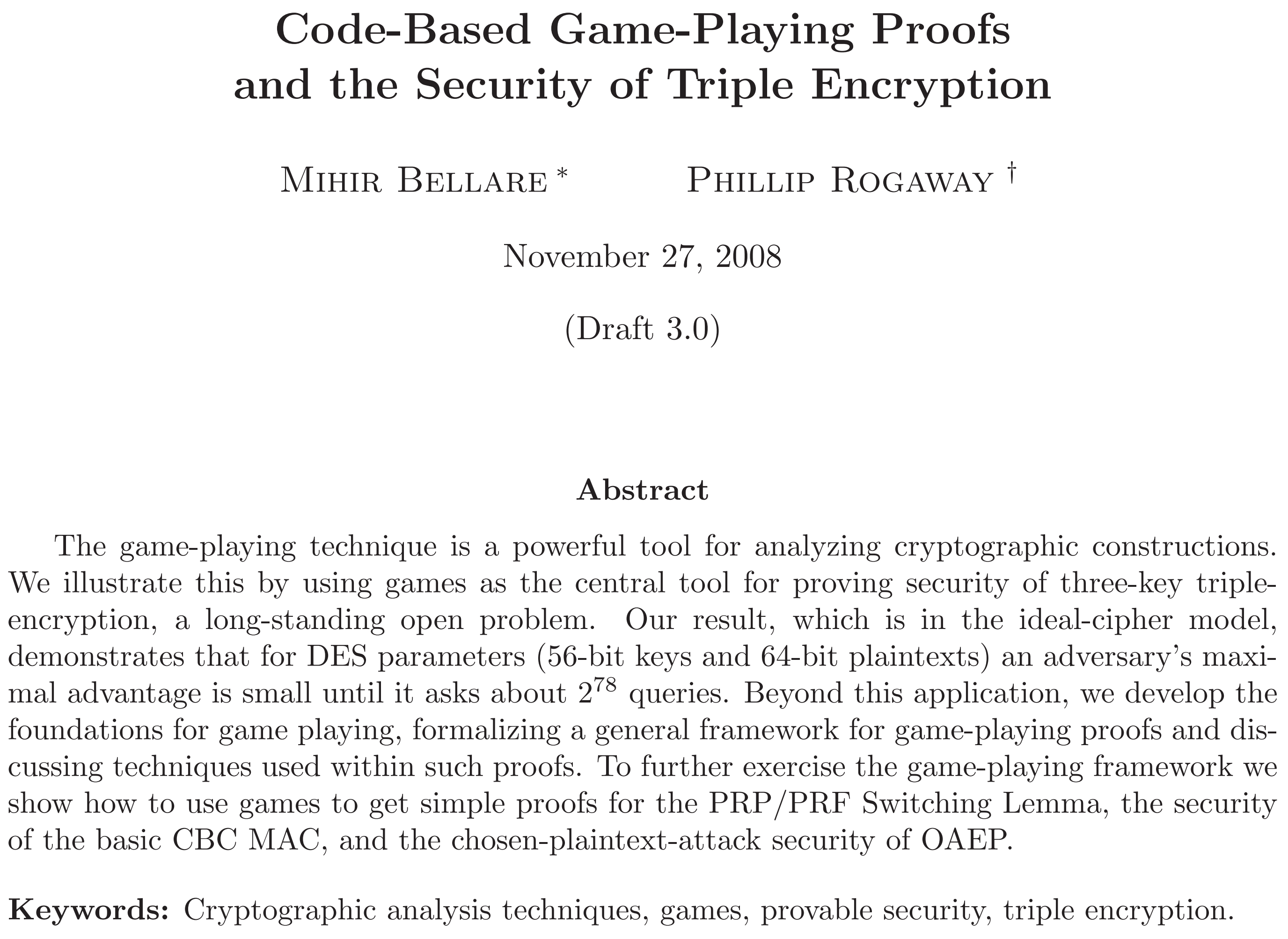
Tomer Ashur[1] and Raluca Posteuca[1]

imec-COSIC, KU Leuven, Leuven, Belgium
[tomer.ashur, raluca.posteuca]@esat.kuleuven.be

Abstract. In this paper we extend the work presented by Ashur and Posteuca in BalkanCryptSec 2018, by designing 0-correlation key-dependent linear trails covering more than one round of DES. First, we design a 2-round 0-correlation key-dependent linear trail which we then connect to Matsui's original trail in order to obtain a linear approximation covering the full DES and 3DES. We show how this approximation can be used for a key recovery attack against both ciphers. To the best of our knowledge, this paper is the first to use this kind of property to attack a symmetric-key algorithm, and our linear attack against 3DES is the first statistical attack against this cipher.

Keywords: linear cryptanalysis, DES, 3DES, poisonous hull

Better Attacks?



Code-Based Game-Playing Proofs and the Security of Triple Encryption

Mihir Bellare [*]   Phillip Rogaway [†]

November 27, 2008

(Draft 3.0)

Abstract

The game-playing technique is a powerful tool for analyzing cryptographic constructions. We illustrate this by using games as the central tool for proving security of three-key triple-encryption, a long-standing open problem. Our result, which is in the ideal-cipher model, demonstrates that for DES parameters (56-bit keys and 64-bit plaintexts) an adversary's maximal advantage is small until it asks about 2^{78} queries. Beyond this application, we develop the foundations for game playing, formalizing a general framework for game-playing proofs and discussing techniques used within such proofs. To further exercise the game-playing framework we show how to use games to get simple proofs for the PRP/PRF Switching Lemma, the security of the basic CBC MAC, and the chosen-plaintext-attack security of OAEP.

Keywords: Cryptographic analysis techniques, games, provable security, triple encryption.

DESX

$$DESX_{KK_1K_2}(M) = K_2 \oplus DES_K(K_1 \oplus M)$$

- Key length $= 56 + 64 + 64 = 184$
- “effective” key length $= 120$ due to a 2^{120} time meet-in-middle attack

Increasing Block-Length?

We will later see that we would also like a blockcipher with longer block-length.

Increasing Block-Length?

We will later see that we would also like a blockcipher with longer block-length.

Increasing Block-Length?

We will later see that we would also like a blockcipher with longer block-length.

This seems much harder to do using DES.

Increasing Block-Length?

We will later see that we would also like a blockcipher with longer block-length.

This seems much harder to do using DES.

Increasing Block-Length?

We will later see that we would also like a blockcipher with longer block-length.

This seems much harder to do using DES.

Motivated the search for a new blockcipher.

AES History

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 Construction

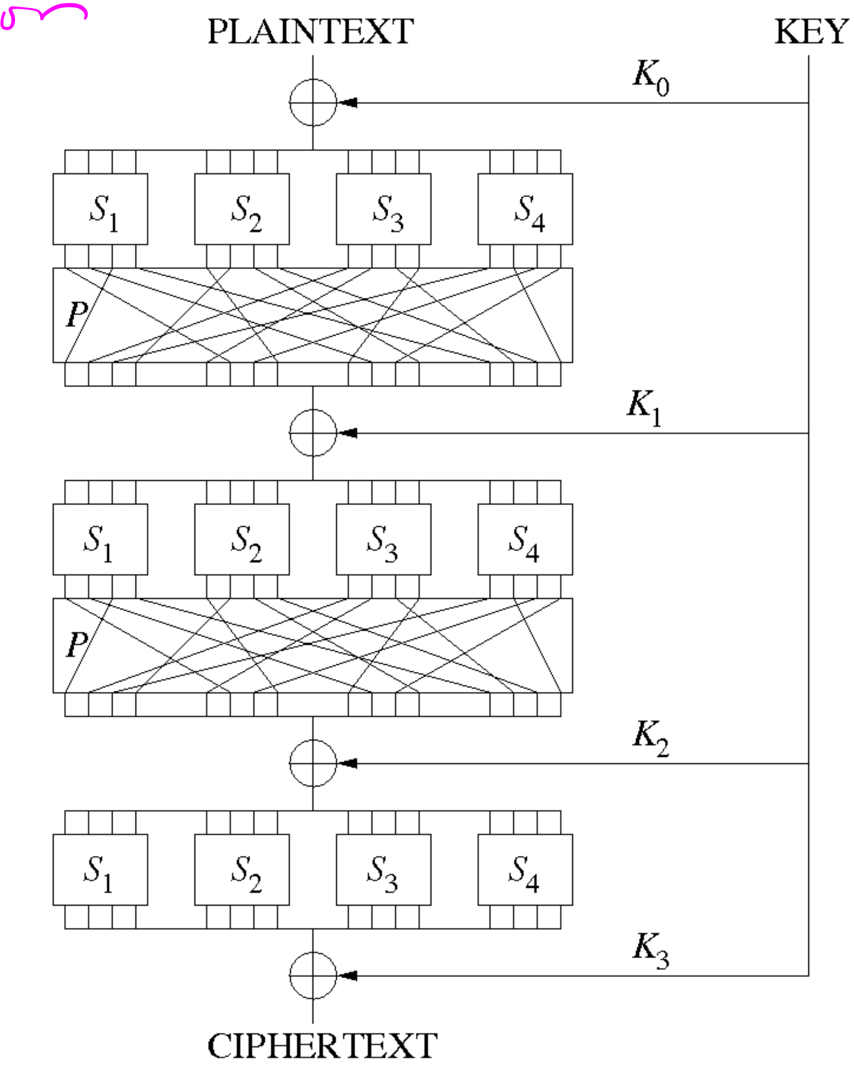
```
function $\mathsf{AES}_K(M)$
    $(K_0, \ldots, K_{10}) \leftarrow \mathit{expand}(K)$
    $s \leftarrow M \oplus K_0$
    for $r = 1$ to $10$ do
        $s \leftarrow S(s)$
        $s \leftarrow \mathit{shift\text{-}rows}(s)$
        if $r \leq 9$ then $s \leftarrow \mathit{mix\text{-}cols}(s)$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

- Fewer tables than DES
- Finite field operations

AES Construction

Substitution permutation network

(vs Feistel rounds)



AES Security

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}.

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 $\mathbf{Adv}_F^{\mathrm{kr}}(A) = 1$ and having running time $\mathcal{O}(2^{128} \cdot T_{\text{AES}})$ where the big-oh hides some small constant.

Is Key-Recovery Security Enough?

No!

Consider ~~identi~~

identity blockciphers :)

2-query EKS : $2^{256} \cdot 4 T_E$ + 2 Fn queries

$$E'_{K_1,K_2}(x_1, x_2) = E_{K_1}(x_1) \| E_{K_2}(x_2)$$

weakness?.. doesn't use Shannon's criteria...

Let $K_1, \dots, K_{2^{128}}$ be an enumeration of the keys.

Adversary A query phase

```
For $i = 1$ to ~~do~~ 2 do:
    $y_{i1} \| y_{i2} \leftarrow$ Fn$(x_{i1} \| x_{i2})$
    // $x_i$ are arbitrary
For $j = 1$ to $2^{128}$ do:
    If $y_{i1} = E_{K_j}(x_{i1})$ $\forall i$
        $K^* \leftarrow K_j$
    If $y_{i2} = E_{K_j}(x_{i2})$ $\forall i$
        $K^{**} \leftarrow K_j$
Ret $K^* \| K^{**}$
```

Crack each key in one loop

$2^{128} \cdot 4 T_E$ + 2 Fn queries

Best KR adversary I can find.