Computer Sciences, UW–Madison — Spring 2012
CS 838: Applied Cryptography — **Instructor:** Thomas Ristenpart
Problem Set 1 Solutions — October 4, 2009

Problem Set 1 Solutions

Problem 2. Let K be a 56-bit DES key, let L be a 64-bit string, and let M be a 64-bit plaintext. Let

$$\mathsf{DESY}(K \| L, M) = \mathsf{DES}(K, L \oplus M)$$
$$\mathsf{DESW}(K \| L, M) = L \oplus \mathsf{DES}(K, M) \,.$$

This defines block ciphers $\mathsf{DESY}, \mathsf{DESW}\colon \{0,1\}^{120} \times \{0,1\}^{64} \to \{0,1\}^{64}$.

Present the best possible key-recovery attacks that you can on these block ciphers. Your attacks should use very few input-output examples, not more than three. State the running time of your attacks.

Note that $C = \mathsf{DESY}(K \| L, M)$ iff $\mathsf{DES}^{-1}(K, C) \oplus M = L$. This leads to the following key-recovery attack:

```
Adversary $A_{\mathsf{DESY}}((M_1, C_1), (M_2, C_2), (M_3, C_3))$
  for each key $T \in \{0,1\}^{56}$ do
    $L_1 \leftarrow \mathsf{DES}^{-1}(T, C_1) \oplus M_1$ ; $L_2 \leftarrow \mathsf{DES}^{-1}(T, C_2) \oplus M_2$ ; $L_3 \leftarrow \mathsf{DES}^{-1}(T, C_3) \oplus M_3$
    if $L_1 = L_2 = L_3$ then return $T \| L_1$
```

The time taken by this attack is that of about $3 \cdot 2^{56}$ DES^{-1} computations.

Note that $C = \mathsf{DESW}(K \| L, M)$ iff $C \oplus \mathsf{DES}(K, M) = L$. This leads to the following key-recovery attack:

```
Adversary $A_{\mathsf{DESW}}((M_1, C_1), (M_2, C_2), (M_3, C_3))$
  for each key $T \in \{0,1\}^{56}$ do
    $L_1 \leftarrow \mathsf{DES}(T, M_1) \oplus C_1$ ; $L_2 \leftarrow \mathsf{DES}(T, M_2) \oplus C_2$ ; $L_3 \leftarrow \mathsf{DES}(T, M_3) \oplus C_3$
    if $L_1 = L_2 = L_3$ then return $T \| L_1$
```

The time taken by this attack is that of about $3 \cdot 2^{56}$ DES computations.

As usual, we are only guaranteed the attacks find a key consistent with the input-output examples rather than finding the target key itself, but empirically we estimate that with three input-output examples the target key will be the only one consistent with the input-output examples and hence will be the one found by the attack. The same attacks using only two input-output examples will also typically find the target key, although perhaps with less frequency than the version using three input-output examples. But if you use only one input-output example, you will almost *never* find the target key. In that case, for every T one computes an L so that $T \| L$ is consistent with the single input-output example, so the attack terminates in one try, but with the wrong key most of the time.

Problem 3. Define the family of functions $F\colon \{0,1\}^{128} \times \{0,1\}^{128} \to \{0,1\}^{128}$ by $F(K,M) = \mathsf{AES}(M,K)$. Assuming AES is a secure PRF, is F a secure PRF? If so, explain why. If not, present the best attack (with analysis) that you can.

F is **not** a secure PRF. The easiest way to see this is to note that it is not even secure against key-recovery: given one input-output example (M,C) of F_K, we can recover K via $K \leftarrow \mathsf{AES}_M^{-1}(C)$. However, this is not enough. The question was whether it is a secure PRF, not whether one can recover the key. To bridge this gap, we can use Proposition 3.14. To this end, first, following Definition 3.12, we formalize the above attack to present the following key-recovery adversary:

```
adversary $B$
Let $M$ be any 128 bit string
$C \leftarrow \mathbf{Fn}(M)$ ; $K \leftarrow \mathsf{AES}_M^{-1}(C)$
Return K
```

Now, looking at Definition 3.12, we see that $\mathbf{Adv}_F^{\mathrm{kr}}(B) = 1$. Now we can apply Proposition 3.14 to conclude that F is not a secure PRF.

An alternative solution is to demonstrate the insecurity of F as PRF directly, by considering the following adversary A that is given an oracle $\mathbf{Fn}\colon \{0,1\}^{128} \to \{0,1\}^{128}$:

```
adversary $A$
Let $M, N$ be any two distinct 128 bit strings
$C \leftarrow \mathbf{Fn}(M)$ ; $L \leftarrow \mathsf{AES}_M^{-1}(C)$
$D \leftarrow \mathbf{Fn}(N)$
if ($\mathsf{AES}(N,L) = D$) then return 1 else return 0
```

We claim that

$$\Pr\left[\mathrm{Real}_F^A \Rightarrow 1\right] \;=\; 1 \quad\text{and}\quad \Pr\left[\mathrm{Rand}_{\{0,1\}^{128}}^A \Rightarrow 1\right] \;=\; 2^{-128}\,.$$

Why? If $\mathbf{Fn} = F_K$ is an instance of F then $C = F(K,M) = \mathsf{AES}(M,K)$, and thus $L = \mathsf{AES}_M^{-1}(C) = K$. Then $D = F(K,N) = \mathsf{AES}(N,K)$, but this equals $\mathsf{AES}(N,L)$, since $L = K$, so A returns 1 with probability one, justifying the first equation above. If $\mathbf{Fn}$ is a random function, then D is distributed uniformly and independently of N, L, and thus the probability that $D = \mathsf{AES}(N,L)$ is 2^{-128}. Now, subtracting, as per Definition 3.6, we get

$$\begin{aligned}\mathbf{Adv}_F^{\mathrm{prf}}(A) &= \Pr\left[\mathrm{Real}_F^A \Rightarrow 1\right] - \Pr\left[\mathrm{Rand}_{\{0,1\}^{128}}^A \Rightarrow 1\right] \\ &= 1 - 2^{-128}\,.\end{aligned}$$

The prf-advantage of our adversary is essentially one. Our adversary is very practical, making just two oracle queries and with running time that of a couple of AES or AES^{-1} computations. So we have a highly effective attack, showing that F is very insecure as a PRF.

Note that the design of A is as in the proof of Proposition 3.14 based on the B we presented above.

Problem 4. Let $F\colon \{0,1\}^k \times \{0,1\}^l \to \{0,1\}^L$ be a family of functions where $l, L \geq 128$. Consider the game G of Fig. 1.

```
main G
$K \xleftarrow{\$} \{0,1\}^k$ ; $b \xleftarrow{\$} \{0,1\}$
$b' \xleftarrow{\$} A^{\mathbf{LR}}$
Ret $(b = b')$

procedure LR($x_0, x_1$)
Ret $F(K, x_b)$
```

Figure 1: Game G for Problem 4.

We define

$$\mathbf{Adv}_F^{\mathrm{lr}}(B) \;=\; 2 \cdot \Pr\left[\mathrm{G}^B \Rightarrow \mathsf{true}\right] - 1 \,.$$

Let $(x_0^1, x_1^1), \ldots, (x_0^q, x_1^q)$ be the queries that B makes to its oracle. (Each query is a pair of l-bit strings, and there are q queries in all.) We say that B is *legitimate* if $x_0^1, \ldots, x_0^q$ are all distinct, and also $x_1^1, \ldots, x_1^q$ are all distinct. We say that F is LR-secure if $\mathbf{Adv}_F^{\mathrm{lr}}(B)$ is "small" for every legitimate B of "practical" resources.

1. Show that the legitimacy condition is necessary for LR-security to be "interesting" by showing that if F is a block cipher then there is an efficient, illegitimate B such that $\mathbf{Adv}_F^{\mathrm{lr}}(B) = 1$. Say how may queries B uses and what is its time-complexity.

Consider the following adversary:

```
adversary B
Let $x, y, z$ be any distinct $l$-bit strings
$C_1 \leftarrow \mathbf{LR}(x, y)$ ; $C_2 \leftarrow \mathbf{LR}(z, y)$
If $C_1 = C_2$ then return 1 else return 0
```

Note B is not legitimate because its queries are of the form $(x_0^1, x_1^1), (x_0^2, x_1^2)$ with $x_1^1 = x_1^2$. Now, if the challenge bit $b = 1$, then $C_1 = F_K(y)$ and $C_2 = F_K(y)$ so $C_1 = C_2$ and B returns 1. (That is, its output b' equals b.) On the other hand if $b = 0$ then $C_1 = F_K(x)$ and $C_2 = F_K(z)$. But since F is a block cipher (this is where we use this assumption) the map F_K is a permutation, and thus $C_1 \neq C_2$. So B returns 0. (That is, its output b is again equal to b.) So $\Pr[b = b'] = 1$, and thus $\mathbf{Adv}_F^{\mathrm{lr}}(B) = 2 \cdot \Pr[b = b'] - 1 = 1$. Adversary B makes only two oracle queries and has time-complexity $O(l + L + T_F)$ where T_F is the time for one evaluation of F. Thus, this is a very practical attack.

2. Let B be a legitimate lr-adversary that makes q oracle queries and has time-complexity t. Show that there exists a prf-adversary A, also making q oracle queries and having time-complexity close to t, such that

$$\mathbf{Adv}_F^{\mathrm{lr}}(B) \;\leq\; 2 \cdot \mathbf{Adv}_F^{\mathrm{prf}}(A) \,. \tag{1}$$

State what is the time-complexity of A. Explain why this reduction shows that if F is a secure PRF then it is LR-secure.

This is very similar to several reductions done in class and the notes. Recall that adversary A gets an oracle for a function $\mathbf{Fn}$: $\{0,1\}^l \to \{0,1\}^L$. It works as follows:

```
adversary A
$b \xleftarrow{\$} \{0,1\}$
$d \xleftarrow{\$} B^{\mathrm{SIM}(\cdot,\cdot)}$
If $d = b$ then return 1 else return 0

procedure SIM($M_0, M_1$)
Ret $\mathbf{Fn}(M_b)$
```

Our adversary picks at random a bit b to represent the challenge bit in game G. It then defines a subroutine SIM via which it responds to oracle queries made by B. Note that SIM uses b and also invokes A's oracle $\mathbf{Fn}$. After learning B's decision d, A tests whether it is correct, meaning equals the challenge bit b. If so, it declares that its oracle is an instance of F, and otherwise it declares its oracle to be random.

Let us now compute $\mathbf{Adv}_F^{\mathrm{prf}}(A)$. First consider Real_F^A, where oracle $\mathbf{Fn}$ is F_K for $K \xleftarrow{\$} \{0,1\}^k$. In this case, the response of $\mathrm{SIM}(M_0, M_1)$ is $F_K(M_b)$, which is exactly $\mathbf{LR}(M_0, M_1)$. This means that

$$\Pr\left[\mathrm{Real}_F^A \Rightarrow 1\right] = \Pr\left[b = d\right] = \frac{1}{2} + \frac{1}{2} \cdot \mathbf{Adv}_F^{\mathrm{lr}}(B) .$$

Now consider $\mathrm{Rand}_{\{0,1\}^L}^A$, where $\mathbf{Fn}$ implements a random function. In this case, the response of $\mathrm{SIM}(M_0, M_1)$ is the random value returned by $\mathbf{Fn}(M_b)$. The legitimacy of B (this is where we use this assumption) now implies that the sequence of responses to B's oracle queries is distributed identically whether $b = 0$ or $b = 1$, in both cases being a sequence of random and independent L bit strings. Thus B gets no information about b from the oracle. This means that $\Pr[b = d] = 1/2$. Thus

$$\Pr\left[\mathrm{Rand}_{\{0,1\}^L}^A \Rightarrow 1\right] = \Pr\left[b = d\right] = \frac{1}{2} .$$

Subtracting, we get

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = \Pr\left[\mathrm{Real}_F^A \Rightarrow 1\right] - \Pr\left[\mathrm{Rand}_{\{0,1\}^L}^A \Rightarrow 1\right] = \frac{1}{2} \cdot \mathbf{Adv}_F^{\mathrm{lr}}(B) ,$$

which implies Equation (1). The time-complexity of A is only $O(1)$ more than that of B given our conventions about measuring time-complexity.

Why does this reduction show that if F is a secure PRF then it is LR-secure? For the usual reason with such reductions. To show that F is LR-secure we need to show that $\mathbf{Adv}_F^{\mathrm{lr}}(B)$ is small for any practical B. However, if B is practical, so is A, and then the assumption that F is a PRF tells us that $\mathbf{Adv}_F^{\mathrm{prf}}(A)$ is small. Then Equation (1) tells us that $\mathbf{Adv}_F^{\mathrm{lr}}(B)$ is also small, as desired.

3. Is the converse true? Namely, if F is LR-secure, then is it a secure PRF? Answer YES or NO. If you say YES, justify this via a reduction, and, if NO, via a counter-example. (The latter means a particular family of functions F which you can prove is LR-secure but which you can show via an attack is not a PRF.)

The answer is NO. A simple counter-example is a family of functions F in which F_K is a constant function for each $K \in \{0,1\}^k$. To be specific, consider the family F defined by $F(K,x) = 0^L$ for all $K \in \{0,1\}^k$ and $x \in \{0,1\}^l$. Here is an attack showing that F is not a PRF:

```
adversary $A$
If $\mathbf{Fn}(0^l) = 0^L$ then return 1 else return 0
```

In game Real_F, the oracle $\mathbf{Fn}$ implements F_K for some K and so $\mathbf{Fn}(0^l) = F_K(0^l) = 0^L$. On the other hand, in game $\text{Rand}_{\{0,1\}^L}$ the probability that $\mathbf{Fn}(0^l) = 0^L$ is 2^{-L}. Thus

$$\Pr\left[\text{Real}_F^A \Rightarrow 1\right] = 1 \quad \text{and} \quad \Pr\left[\text{Rand}_{\{0,1\}^L}^A \Rightarrow 1\right] = 2^{-L} .$$

Subtracting, we get $\mathbf{Adv}_F^{\text{prf}}(A) = 1 - 2^{-L}$. This is close to 1 (recall the problem assumes $L \geq 128$) and furthermore A is very efficient, so we have shown that F is not a PRF. On the other hand, we claim that F is LR-secure. To see this consider any B with a $\mathbf{LR}(\cdot,\cdot)$ oracle. The response of the oracle to any query is 0^L. In particular, this is true regardless of the value of b, meaning the oracle responses give B no information about b. Thus $\Pr[b = d] = 1/2$, where d is the output of B, and so $\mathbf{Adv}_F^{\text{lr}}(B) = 0$. So we have shown LR-security in a very strong sense: the advantage of any adversary is 0, regardless of its time-complexity or the number of queries it makes.

We clarify that F above is a family of functions. It is not required to be a block cipher except in part **1.**

Extra credit

The goal of a key-search attack (such as exhaustive key search) is to find the target key, but, as discussed in the notes and in class, such an attack might find a key that is consistent with the input-output examples but is not the target key. We glossed over this, saying it "usually" does not happen. This problem gives a sense of how cryptographers arrive at this type of conclusion and estimate what "usually" means.

We use what is called the *ideal cipher model.* Let $k, n \geq 1$ be integers. Let $K = 2^k$ and $N = 2^n$ and let $T_1, \ldots, T_K$ be some enumeration of the elements of $\{0,1\}^k$. We consider a thought experiment in which a block cipher is chosen at random. By this we mean that for each key T_i, we choose $E(T_i, \cdot)$ as a random permutation on $\{0,1\}^n$. Fix a message $M^* \in \{0,1\}^n$ known to the adversary, who, given a ciphertext $C^* = E(T^*, M^*)$ for a random, unknown T^* attempts to find T^*. The adversary can access E (only) as an oracle.

We formalize this via the game EKS of Fig. 2. We will use games a lot so this is a good chance to start getting familiar with them. The game maintains a table E, representing the block cipher, and assumed to initially be $\perp$ (undefined) everywhere. It also associates to each key T a set $\text{Range}[T]$ that is initially empty. The game is executed with an adversary A. As this execution continues, the tables get populated, and the block cipher gets slowly defined. First, the main procedure executes. It picks a random challenge key T^*, defines $E[T^*, M^*]$ to be a random n-bit string, and returns it to the adversary as the challenge ciphertext C^*. Now the adversary executes, and can make queries

```
main EKS
$T^* \xleftarrow{\$} \{0,1\}^k$ ; $C^* \leftarrow E[T^*, M^*] \xleftarrow{\$} \{0,1\}^n$
Range$[T^*] \leftarrow \{C^*\}$
$T \xleftarrow{\$} A^{\mathbf{E}}(C^*)$
Ret $(T = T^*)$

procedure $\mathbf{E}(T, M)$
If not $E[T,M]$ then $E[T,M] \xleftarrow{\$} \{0,1\}^n \setminus$ Range$[T]$
Range$[T] \leftarrow$ Range$[T] \cup \{E[T,M]\}$
Return $E[T,M]$
```

Figure 2: Game EKS for Problem 5.

of the form T, M to procedure $\mathbf{E}$. A query T, M creates the point $E[T, M]$. It is chosen at random, but, to ensure the permutation property of a block cipher, from the set

$$\{0,1\}^n \setminus \mathrm{Range}[T] = \{0,1\}^n \setminus \{\, E[T, M'] \;:\; E[T, M'] \neq \bot \,\} \,.$$

The test "If not $E[T, M]$" returns true iff $E[T, M]$ is undefined, meaning equal to $\bot$ rather than an n-bit string. The set Range$[T]$ contains all points $E[T, M]$ that are currently defined. When the adversary is done, it outputs its guess T for the value of T^*. The game returns true if $T = T^*$ and false otherwise. The output of main is called the output of the game or execution, and we let $\Pr[\mathrm{EKS}^A]$ denote the probability that this output is true. The probability is over the random choices in the game, as well as those of the adversary, if any.

Here we are considering a very simple form of key search where there is only one input-output example.

Now, using this model, we can try to calculate the probability that an attack returns the target key, as opposed to some non-target key consistent with the input-output examples.

Problem 5. Let $k, n \geq 1$ be integers. Let $K = 2^k$ and $N = 2^n$. Fix $M^* \in \{0,1\}^n$ and let $T_1, \ldots, T_K$ be some enumeration of the elements of $\{0,1\}^k$. Consider the following adversary for game EKS:

```
adversary $A(C^*)$
For $i = 1, \ldots, K$ do
    If $\mathbf{E}(T_i, M^*) = C^*$ then $G \leftarrow T_i$ ;  return $G$
```

This adversary calls the $\mathbf{E}$ oracle up to K times as shown. Let $\mathbf{Adv}^{\mathrm{eks}}(K, N) = \Pr[\mathrm{EKS}^A]$. This is the probability that the key G output by A in its execution with EKS equals the target key T^* chosen by main.

1. Prove that

$$\mathbf{Adv}^{\mathrm{eks}}(K, N) \;=\; \frac{N}{K} \cdot \left[1 - \left(1 - \frac{1}{N}\right)^K\right] \,. \tag{2}$$

We justify the following chain of equalities below:

$$
\begin{aligned}
& \mathbf{Adv}^{\mathrm{eks}}(K,N) && (3)\\
&= \Pr\left[G = T^*\right] \\
&= \sum_{j=1}^{K} \Pr\left[G = T^* \wedge T^* = T_j\right] && (4)\\
&= \sum_{j=1}^{K} \Pr\left[G = T^* \mid T^* = T_j\right] \cdot \Pr\left[T^* = T_j\right] && (5)\\
&= \sum_{j=1}^{K} \Pr\left[G = T^* \mid T^* = T_j\right] \cdot \frac{1}{K} && (6)\\
&= \frac{1}{K} \cdot \sum_{j=1}^{K} \Pr\left[G = T^* \mid T^* = T_j\right] \\
&= \frac{1}{K} \cdot \sum_{j=1}^{K} \Pr\left[E[T_1,M^*] \neq C^* \wedge E[T_2,M^*] \neq C^* \wedge \ldots \wedge E[T_{j-1},M^*] \neq C^*\right] && (7)\\
&= \frac{1}{K} \cdot \sum_{j=1}^{K} \prod_{i=1}^{j-1} \Pr\left[E[T_i,M^*] \neq C^*\right] && (8)\\
&= \frac{1}{K} \cdot \sum_{j=1}^{K} \prod_{i=1}^{j-1} \left(1 - \frac{1}{N}\right) && (9)\\
&= \frac{1}{K} \cdot \sum_{j=1}^{K} \left(1 - \frac{1}{N}\right)^{j-1} \\
&= \frac{1}{K} \cdot \sum_{j=0}^{K-1} \left(1 - \frac{1}{N}\right)^{j} \\
&= \frac{1}{K} \cdot \frac{1 - \left(1 - \frac{1}{N}\right)^K}{1 - \left(1 - \frac{1}{N}\right)} && (10)\\
&= \frac{N}{K} \cdot \left[1 - \left(1 - \frac{1}{N}\right)^K\right] .
\end{aligned}
$$

We now justify the numbered steps. (The others are straightforward.)

Equation (4) is true because the events $T^* = T_j$ $(1 \leq j \leq K)$ are a partition of the underlying probability space, meaning exactly one (not less, not more) of them is always true.

Equation (5) uses the definition of conditional probability, namely $\Pr\left[A \mid B\right] = \Pr[A \wedge B]/\Pr[B]$.

Since T^* is chosen at random, the probability that $T^* = T_j$ is $1/K$ for any j, which justifies Equation (6).

For Equation (7), fix some j and assume $T^* = T_j$. Now what is the probability that the attack returns $G = T_j$? Look at the code. It must be the case that $E[T_i, M^*] \neq C^*$ for all $i = 1, \ldots, j-1$, since otherwise the code will return T_i rather than T_j. (In other words, the $j-1$ keys preceding T_j must not yield false positives.) Once this is true, the loop index i arrives at the value j. At that point, the code will certainly halt and return T_j, because the test $E[T_j, M^*] = C^*$ is true. (Because $T_j = T^*$.) Thus, $\Pr[G = T^* \mid T^* = T_j]$ is exactly the probability that $E[T_i, M^*] \neq C^*$ for all $i = 1, \ldots, j-1$. (Note that whether or not $E[T_l, M^*] = C^*$ for $l > j$ is not relevant because the code halts and returns when $i = j$.)

Equation (8) is true because the events $E[T_i, M^*] = C^*$ are independent for $i = 1, \ldots, j-1$. This is true because of the random choices made in defining E and because $T_1, \ldots, T_{j-1}$ are distinct.

The random choices that define E ensure that $\Pr[E[T_i, M^*] = C^*] = 1/N$ for all $i = 1, \ldots, j-1$, which justifies Equation (9).

Equation (10) uses the standard formula $\sum_{i=0}^{K-1} a^i = (1-a^K)/(1-a)$ for the sum of a geometric series, with $a = 1 - 1/N$.

That completes the proof.

2. It is difficult to get a quantitative feel from Equation (2). We will now lower bound it via a simpler expression. To do so we first recall an inequality. Namely let x be a real number in the range $0 \leq x \leq 1$. Let m, l be integers such that $0 \leq l \leq m$ and l *is even*. Then

$$(1-x)^m \;\leq\; \sum_{i=0}^{l} \binom{m}{i} (-x)^i \,. \tag{11}$$

Use this and the result of **1.** above to show that

$$\mathbf{Adv}^{\text{eks}}(K, N) \;\geq\; 1 - \frac{K-1}{2N} \,. \tag{12}$$

We will use the given inequality with $x = 1/N$, $m = K$ and $l = 2$. This gives us

$$\begin{aligned}
\left(1 - \frac{1}{N}\right)^K &\leq \binom{K}{0}\left(\frac{-1}{N}\right)^0 + \binom{K}{1}\left(\frac{-1}{N}\right)^1 + \binom{K}{2}\left(\frac{-1}{N}\right)^2 \\
&= 1 - \frac{K}{N} + \frac{K(K-1)}{2N^2} \,.
\end{aligned}$$

Now from **1.** above we have

$$\begin{aligned}
\mathbf{Adv}^{\text{eks}}(K, N) &= \frac{N}{K} \cdot \left[1 - \left(1 - \frac{1}{N}\right)^K\right] \\
&\geq \frac{N}{K} \cdot \left[1 - \left(1 - \frac{K}{N} + \frac{K(K-1)}{2N^2}\right)\right] \\
&= \frac{N}{K} \cdot \left[\frac{K}{N} - \frac{K(K-1)}{2N^2}\right] \\
&= 1 - \frac{K-1}{2N} \,.
\end{aligned}$$

3. Let k, n be (respectively) the key-length and block-length parameters of DES. Use the result of **2.** to numerically estimate $\mathbf{Adv}^{\mathrm{eks}}(K, N)$ in this case. Do the same when k, n are the parameters of AES.

The DES parameters are $k = 56$ and $n = 64$. In this case we get

$$\mathbf{Adv}^{\mathrm{eks}}(2^{56}, 2^{64}) \;\geq\; 1 - \frac{2^{56} - 1}{2^{65}} \approx 1 - 2^{-9} \;.$$

The AES parameters are $k = 128$ and $n = 128$. In this case we get

$$\mathbf{Adv}^{\mathrm{eks}}(2^{128}, 2^{128}) \;\geq\; 1 - \frac{2^{128} - 1}{2^{129}} \approx \frac{1}{2} \;.$$

Thus with DES parameters, the attack finds the target key except with the relatively small probability of 2^{-9}, while with AES parameters the attack finds the target key less often, namely about half the time.

4. What do these results tell us about the success probability of an exhaustive key-search attack on DES? What about on AES? Is DES an ideal cipher? Is AES an ideal cipher? Discuss.

Let's begin with the last two questions. Is AES "an ideal cipher"? Is DES "an ideal cipher"? To answer these, we would first have to know the meaning or definition of the phrase in quotes. In particular, meaningfully answering such a question pre-supposes a definition of the form: "A block cipher E is said to be an ideal cipher if X is true" where X is some condition on E. Then, we can ask if DES or AES meet condition X.

The difficulty is that a definition as indicated above, and in particular the definition of X, were never given. We never gave any definition of what it meant for a block cipher E to "be an ideal cipher." That is, nowhere do you read a definition of the form: "Let E be a block cipher. Then we say that E is ideal if ...." What we defined instead is an ideal cipher *model*, which involves a game in which a block cipher is chosen at random.

To better understand the difference, suppose our questions had been: Is AES a PRF? Is DES a PRF? In this case, the questions are meaningful, because we did give definitions of the form "Let E be a block cipher. Then we say that E is a PRF if ...". (The condition "..." was given in class and can also be found in the notes.) Now, is the answer yes or no? Well, we don't know. We conjecture it is yes (up to the limitations given by known attacks) but it could be no. But in any case, this is not the question we were asked, so let us return to the discussion.

What then do these ideal-cipher model results tell us about the success probability of an exhaustive key-search attack on DES or AES? Well, speaking strictly mathematically, nothing at all.

So why do we use this model? Because, heuristically, it tells us something. We "believe" that DES and AES have some "properties of an ideal cipher." What we mean by this is that if we figure out the probability of some event in the ideal cipher model, for example the success probability of an attack as above, then in practice, when the attack is run on DES or AES, the success probability will be about what is predicted by the ideal cipher model. We believe this because, heuristically, the block ciphers are designed to have "random behavior."

In principle we might substantiate this belief by experiments. That may be possible for DES, but not for AES, where exhaustive key search attacks are out of practical reach.

Thus it is important to appreciate both the value and the limitations of the ideal cipher model. (Later we will see another model, the random oracle model, with the same features.) If you face the problem of designing or analyzing a block cipher based scheme, it is actually a good idea to first get some sense of what the security would be like in the ideal cipher model. Heuristically you would guess the same would be true if you used a "good" block cipher.

But even better is to use the PRF/PRP model, and that is what we will be doing. (Can you understand why it is better?)
