LASEC
SECURITY AND CRYPTOGRAPHY LABORATORY

EPFL / I & C
CH-1015 Lausanne
Phone: ++41 (0) 21 693 76 03
Fax: ++41 (0) 21 693 68 79
URL: http://lasecwww.epfl.ch

Solution of Midterm

1 Exhaustive Search on 3DES

1. The algorithm successively tries every possible key. It does not stop until the last possible key is tried. Therefore, the number of iterations is exactly equal to the number of possible keys times the number of DES encryption for each (which is 3). Therefore, the number of DES encryption/decryption of the algorithm is $3 \cdot 2^{56*3} = 3 \cdot 2^{168}$.

2. The random permutation C^* is uniformly distributed among all possible permutations, and there are $(2^{64})!$ of them. Therefore, if $\mathsf{c} : \{0,1\}^{64} \to \{0,1\}^{64}$ is a given permutation we have $\Pr[\mathsf{C}^* = \mathsf{c}] = \frac{1}{(2^{64})!}$. Now, we are given two (fixed) values $P, C \in \{0,1\}^{64}$. We have

$$\begin{aligned} \Pr[\mathsf{C}^*(P) = C] &= \sum_{\mathsf{c}} \mathbf{1}_{\mathsf{C}^*(P)=C} \Pr[\mathsf{C}^* = \mathsf{c}] \\ &= \frac{1}{(2^{64})!} \sum_{\mathsf{c}} \mathbf{1}_{\mathsf{C}^*(P)=C} \ , \end{aligned}$$

   where the last sum simply is the number of permutations mapping P on C, which is the number of permutations of a set of cardinality $2^{64} - 1$. Finally,

$$\Pr[\mathsf{C}^*(P) = C] = \frac{(2^{64}-1)!}{(2^{64})!} = 2^{-64} \ .$$

3. We suppose that $\Pr_K[3\mathsf{DES}_K(P) = C] = \Pr_{\mathsf{C}^*}[\mathsf{C}^*(P) = C] = 2^{-64}$. Therefore the number N of keys displayed by the algorithm is

$$\begin{aligned} N &= \Pr_K[3\mathsf{DES}_K(P) = C] \times \#\{\text{number of tried keys}\} \\ &= 2^{-64} \cdot 2^{168} \\ &= 2^{104} \ . \end{aligned}$$

   All the displayed keys (except 1) are wrong keys!

Algorithm 1 Exhaustive key search algorithm on 3DES, using t plaintext/ciphertext couples

Require: t plaintext/ciphertext couples (P_i, C_i), for $i = 1, \ldots, t$, all encrypted under the same key k.

```
for each possible key K = (K_1, K_2, K_3) do
    if C_i = 3DES_K(P_i) for i = 1, ..., t then
        display K = (K_1, K_2, K_3)
    end if
end for
```

4. We consider Algorithm 1. The algorithm clearly displays k as we do have $C_i = 3\mathsf{DES}_k(P_i)$ for all $i = 1, \ldots, t$. It does reduce the number of wrong keys that are displayed because it clearly is more difficult to find a wrong key $\widetilde{k}$ that $C_i = 3\mathsf{DES}_{\widetilde{k}}(P_i)$ for $i = 1, \ldots, t$ (with $t > 1$) than to find a wrong key such that $C = 3\mathsf{DES}_{\widetilde{k}}(P)$ (for only one couple). The total number of encryption/decryption steps that have to be performed is simply t times the number found in the first question (we suppose that we always perform t 3DES in the **if** statement of the algorithm). Therefore, this algorithm needs $3 \cdot 2^{168} \cdot t$ encryptions/decryptions.

5. Still supposing that $\Pr_K[3\mathsf{DES}_K(P) = C] = \Pr_{\mathsf{C}^*}[\mathsf{C}^*(P) = C] = 2^{-64}$, the mean value N of wrong keys displayed by Algorithm 1 is

$$\begin{aligned} N &= \#\{\text{number of tried keys}\} \times \prod_{i=1}^{t} \Pr_K[3\mathsf{DES}_K(P_i) = C_i] \\ &= 2^{168} \cdot (2^{-64})^t \, . \end{aligned}$$

Table 1 gives the approximate number N of wrong keys that are displayed, in function of the number t of plaintext/ciphertext couples that are available. According to this table, only 3 couples are necessary to make almost sure that only the good key will be displayed.

Table 1: Mean value N of wrong keys that are displayed by Algorithm 1, in function of t

t	1	2	3
N	2^{104}	2^{40}	2^{-24}

2 Multicollisions on Hash Functions

Preliminaries

1. According to the birthday paradox, we need approximatively $2^{n/2}$ messages to find a collision on h (i.e., a 2-collision on h) with a probability of success of $1 - e^{-1/2} \approx 0.393$.

Multicollisions in Iterated Hash Functions

2. Using the birthday paradox once again, in order to find a collision on the compression function, we need $\theta \cdot 2^{n/2}$ blocks in order to find a collision, with a probability of success of $1 - e^{-\frac{\theta^2}{2}}$. As the block are chosen in a set of cardinality $2^\ell \gg 2^{n/2}$, there are enough of them to be sure to find a collision.

3. The idea is to search for two distinct blocks B_1 and B_1' such that $f(\text{IV}, B_1) = f(\text{IV}, B_1')$. Calling x_1 this output of the compression function, we then search for B_2 and B_2' such that $f(x_1, B_2) = f(x_1, B_2')$. We call x_2 this last value. This is represented on Figure 1. We now

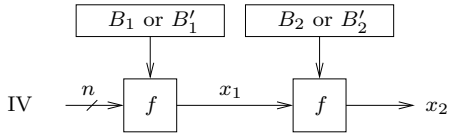

Figure 1: How to find a 4-collision on h_0

consider the four following messages: $B_1 \parallel B_2$, $B_1 \parallel B_2'$, $B_1' \parallel B_2$, and $B_1' \parallel B_2'$. Clearly, the all produce the same hash value y when they are hashed with h_0. Therefore, we have found a 4-collision on h_0. In order to do this, we had to find two 2-collision on the compression function f, so that the overall complexity is $2 \cdot \theta \cdot 2^{n/2}$, for a probability of success of $(1 - e^{-\theta^2/2})^2$ (as we need both collision searches to be successful).

4. Suppose we hash the four messages of the preceeding question with h instead of h_0. The only difference is that a padding has to be concatenated to the messages. But as this padding only depends on the length of the message to be hashed, all four messages will have the same padding (that we denote PAD). We represent this situation on Figure 2.

5. We denote $\text{IV} = x_0$ and construct the sequence x_i, $i = 1, \ldots, t$ as follows. Given x_{i-1}, find two distinct blocks B_i and B_{i-1} such that

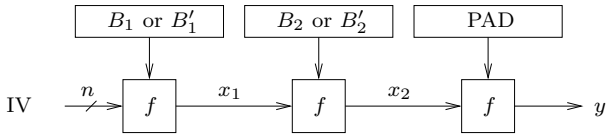

Figure 2: How to find a 4-collision on h, based on a 4-collision on h_0

$f(x_{i-1}, B_i) = f(x_{i-1}, B'_i)$. This corresponds to a 2-collision search on f. Call x_i this value. This construction is represented on Figure 3. Clearly, the 2^t messages $\{B_1, B'_1\} \parallel \{B_2, B'_2\} \parallel \cdots \parallel \{B_t, B'_t\}$ all

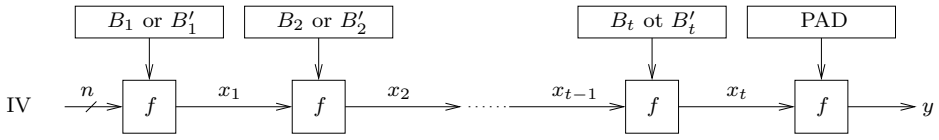

Figure 3: Finding a 2^t-collision on h

produce the same h_0 hash value. As they all are of the same length (t blocks) this implies that they also produce the same h value. We have obtained a 2^t-collision on h.

6. According to the preceeding question, we need t successful collision searches on f. If we make $\theta \cdot 2^{n/2}$ calls to f each time we look for a collision on f, this makes a total of $t \cdot \theta \cdot 2^{n/2}$ calls to f. We need the t collision searches to be successful, so that the overall probability of success is $(1 - e^{-\theta^2/2})^t$.

Multicollisions in the Random Oracle Model

7. The number of functions from $\mathcal{M}$ to $\mathcal{H}$ is $\#\mathcal{H}^{\#\mathcal{M}}$. We have

$$\begin{aligned} \Pr[H(m_1) = h_1] &= \sum_{\mathsf{h}} \mathbf{1}_{\mathsf{h}(m_1)=h_1} \Pr[H = \mathsf{h}] \\ &= \frac{1}{\#\mathcal{H}^{\#\mathcal{M}}} \sum_{\mathsf{h}} \mathbf{1}_{\mathsf{h}(m_1)=h_1} \ , \end{aligned}$$

where the last sum is the number of functions mapping m_1 on h_1, which is the number of functions of a set of cardinality $\#\mathcal{M} - 1$ to a set of cardinality $\#\mathcal{H}$. Therefore

$$\begin{aligned} \Pr[H(m_1) = h_1] &= \frac{\#\mathcal{H}^{\#\mathcal{M}-1}}{\#\mathcal{H}^{\#\mathcal{M}}} \\ &= \frac{1}{\#\mathcal{H}} \ . \end{aligned}$$

Similarly,

$$\begin{aligned}\Pr[H(m_1)=h_1, H(m_2)=h_2] &= \sum_{\mathsf{h}} \mathbf{1}_{\mathsf{h}(m_1)=h_1,\mathsf{h}(m_2)=h_2} \Pr[H=\mathsf{h}] \\ &= \frac{1}{\#\mathcal{H}\#\mathcal{M}} \sum_{\mathsf{h}} \mathbf{1}_{\mathsf{h}(m_1)=h_1,\mathsf{h}(m_2)=h_2} \ ,\end{aligned}$$

where the last sum is the number of functions mapping m_1 on h_1 and m_2 on h_2, which is the number of functions of a set of cardinality $\#\mathcal{M}-2$ to a set of cardinality $\#\mathcal{H}$. Therfore

$$\Pr[H(m_1)=h_1, H(m_2)=h_2] = \frac{1}{\#\mathcal{H}^2} \ .$$

This proves that $\Pr[H(m_1)=h_1, H(m_2)=h_2] = \Pr[H(m_1)=h_1]\Pr[H(m_2)=h_2]$. Therefore, the two events are independent.

8. Using the lemma with $r=2$, we see that there is not s-coincidence for any $s \geq 2$ in $\{H_1,\ldots,H_q\}$ with a probability e^{λ}, where λ is such that $q=\sqrt{2\lambda}\cdot 2^{n/2}$. Let $\theta=\sqrt{2\lambda}$. In other words, we do have at least a 2-coincidence (a collision) with probability $1-e^{\theta^2/2}$ in $\{H_1,\ldots,H_q\}$, when $q=\theta\cdot 2^{n/2}$.

9. A r-collision in $\{m_1,\ldots,m_q\}$ corresponds to a r-coincidence in $\{H_1,\ldots,H_q\}$. We obtain (at least) a r-coincidence with probability $1-e^{-1/2}$ (i.e., $\lambda=\frac{1}{2}$) when $q=(\frac{r!}{2})^{1/r}2^{n(r-1)/r}$.

10. With $r=4$ and $n=128$, the preceeding relation gives $q=12^{1/4}2^{96} > 2^{96}$. For iterated hash function, we showed that a 4-collision can be found with probability $(1-e^{-\theta^2/2})^2 \approx 1-2\cdot e^{-\theta^2/2}$ when $q=2\cdot\theta\cdot 2^{n/2}$. This shows that we roughly need 2^{64} hash computations for similar probabilities of success. This is indeed much smaller than 2^{96}.

11. We can see that the values found in the random oracle model are way larger than the realistic ones. The random oracle model is definitively of no help for studying this problem!