Assignment 3: Block Ciphers

CSCI3381-Cryptography

Due October 3, 2014

1 Solutions to the Written Problems

1. Block Cipher Modes of Operation *6 points per part, 30 total.* Parts (a)-(d) refer to the cipherblock chaining mode (CBC), and part (e) is about a variant. These problems do not depend at all on the specific block cipher used. Explain your answers carefully.

(a) Suppose a message of 100 plaintext blocks is being encrypted with CBC mode. Suppose that, prior to encryption, one bit of the tenth plaintext block is inadvertently changed. How many blocks of plaintext, after decryption, are certain to be correct?

(b) Same setup as part (a), but this time, one bit of the tenth *ciphertext* block is changed after encryption but before decryption. How many blocks of plaintext, after decryption, are certain to be correct?

(c) Same set up as (b), but this time the tenth and eleventh ciphertext blocks are swapped before decryption.

Solution for parts (a)-(c) The first three questions can all be answered by keeping in mind that on encryption, the i^{th} ciphertext block depends on *all* the preceding plaintext blocks, but on decryption, the i^{th} plaintext block depends only on ciphertext blocks i and $i-1$. The details:

We have

$$C_i = E_K(P_i \oplus C_{i-1})$$

for all $i \geq 0$ (treating the IV as cipher block 0), and thus

$$P_i = D_K(C_i) \oplus C_{i-1}.$$

Part (a): If P_i is unchanged for $i < 10$, but P_{10} is changed in a single bit, then $P_{10} \oplus C_9$ will be also change one bit, and $C_{10} = E_K(P_{10} \oplus C_9)$ will be completely garbled, as will all subsequent ciphertext blocks. BUT, we have correctly encrypted the sequence

$$P_1, P_2, \ldots, P'_{10}, P_{11}, \ldots P_{100},$$

where P'_{10} denotes the altered tenth block, so that upon decryption 99 blocks of plaintext will still be correct.

Part(b): Changing one bit of C_{10} causes $D_K(C_{10})$ to be completely garbled, and thus $P_{10} = D_K(C_{10}) \oplus C_9$ will also be garbled. $D_K(C_{11})$, however, will be the same, and

thus P_{11} will change in only one bit. Thereafter, all subsequent plaintext blocks will be correct. So 98 blocks of plaintext will still be correct.
Part (c): works just like Part (b): changing cipher block 10 will alter blocks 10 and 11 of plaintext, and then changing block 11 will further alter block 11 and block 12. So we should expect no more than three blocks of plaintext to change. Is there anything in the fact that these blocks are switched that will affect the result? We have, in the original version:

$$P_{10} = D_K(C_{10}) \oplus C_9, P_{11} = D_K(C_{11}) \oplus C_{10}, P_{12} = D_K(C_{12}) \oplus C_{11}.$$

But as a result of the switch, we instead will get

$$P_{10} = D_K(C_{11}) \oplus C_9, P_{11} = D_K(C_{10}) \oplus C_{11}, P_{12} = D_K(C_{12}) \oplus C_{10}.$$

There's no reason to conclude that any of these three blocks is unchanged as a result of the swap. So 97 blocks will be correct.

(d) Suppose we encrypt two multi-block messages with CBC mode but use the same IV each time. What information about the plaintext is leaked by the ciphertext? (You also might try to imagine a scenario in which this information is useful to an attacker.)

Solution. Let P_1, P_1', C_1, C_1' denote the first plaintext and ciphertext blocks of the two messages. With the IV unchanged, we have

$$C_1 = E_K(IV \oplus P_1), C_1' = E_K(IV \oplus P_1').$$

It follows that $P_1 = P_1'$ if and only if $C_1 = C'1$. The attacker can thus tell whether or not two different messages begin with identical first blocks.

(e) Here is a variant of CBC mode called 'Infinite Garble Extension' (I don't know why!) As usual, the plaintext blocks are denoted $P_1, P_2, \ldots,$ the ciphertext blocks $C_1, C_2, \ldots,$ and an initialization vector block IV. The encryption algorithm is given by

$$C_1 = E_K(P_1) \oplus IV,$$
$$C_i = E_K(P_i \oplus C_{i-1}) \oplus P_{i-1}.$$

Write analogous equations defining the decryption operation.

Solution. We just do some $\oplus$ algebra to solve the two equations:

$$P_1 = D_K(C_1 \oplus IV),$$

$$P_i = D_K(C_i \oplus P_{i-1}) \oplus C_{i-1}.$$

In other words, you need both the current ciphertext block, the preceding ciphertext block, *and the preceding plaintext block* to recover the present plaintext block. This means that decryption of the blocks has to be done in first-to-last order, in contrast to CBC, where the cipher blocks can be decrypted in any order.

2. Modifications to Substiution-Permutation Networks. *Point values: 7,8,10; 25 total.* This problem is about the internal structure of block ciphers. The model here is the generic substitution-permutation network described in the notes. (While you don't need to explicitly consider AES in this problem, you should be aware that conclusions also apply to AES, where the mixing permutation is replaced by multiplication by an invertible matrix. Problem 4 of Chapter 5 is relevant to part (c) below.)

(a) Suppose our encryption algorithm uses only *one round* of the SP network. Explain how we can completely decrypt any block given a single known plaintext-ciphertext pair of blocks. Describe the procedure in detail. Does this attack recover the key?

Solution. Let's write each round of the encryption algorithm as

$$Y = \pi(S(X \oplus K)),$$

where X is the input block to the round, Y the output, K the round key, π the mixing permutation, and S the map you get from applying the S-boxes in parallel to the segments of a block. In the case of only one round, this is the entire encryption algorithm. Now remember that π and S are known to the attacker, and since these are maps given by small tables, the attacker can easily compute the inverse maps π^{-1} and S^{-1}. You then get

$$K = S^{-1}(\pi^{-1}(Y)) \oplus X,$$

so that the attacker recovers the key K from a single know plaintext-ciphertext pair.

*(b)*Suppose that instead of alternating AddKey, Sub, and Mix in the r rounds of our substitution cipher, we do all r rounds of AddKey, then all r rounds of substitution, then all r rounds of the mixing permutation, in that order. Show that if we have a single known plaintext-ciphertext pair of blocks, we can decrypt any block. Does this attack recover the key?

Solution. Let's keep the same notation, but run it for three rounds, without alternation. (There's nothing special about three rounds—once you see this solution you will see how it works for any number of rounds.)

$$Y = \pi(\pi(\pi(S(S(S(X \oplus K_1 \oplus K_2 \oplus K_3)))))).$$

The solution is the same as above, applying all the inverse maps and XOR ing to recover, not the original key, but the exclusive or $K_1 \oplus K_2 \oplus K_3$ of the round keys. But this XOR is all you need to encrypt and decrypt messages, so knowing a single plaintext-ciphertext pair breaks the cipher.

(c) Suppose we got rid of the S-boxes, and just retained the AddKey and Mix phases of each round. Show how, given a single known plaintext-ciphertext pair of blocks, we can decrypt any other block of ciphertext. HINT: Show that this forces

$$E_K(M_1) \oplus E_K(M_2) = \pi(M_1 \oplus M_2),$$

where π is some permutation of the bits, for any two plaintext blocks M_1 and M_2. Does this attack recover the key?

Solution. Again we'll keep the notation above, and to be concrete consider three rounds. The encryption algorithm is now

$$E_K(M) = \pi(K_3 \oplus \pi(K_2 \oplus \pi(K_1 \oplus X))).$$

The point is that any permutation like the mixing permutation π is *linear*—that is, it satisfies $\pi(X_1 \oplus X_2) = \pi(X_1) \oplus \pi(X_2)$. So

$$E_K(M) = \pi(K_3) \oplus \pi(\pi(K_2)) \oplus \pi(\pi(\pi(K_1))) \oplus \pi(\pi(\pi(M))).$$

If we let $\pi' = \pi \circ \pi \circ \pi$ we get the result above, that

$$E_K(M_1) \oplus E_K(M_2) = \pi'(M_1) \oplus \pi'(M_2) = \pi'(M_1 \oplus M_2).$$

Thus if we know M_1 and $E_K(M_1)$, and intercept the ciphertext $E_K(M_2)$, we can compute $\pi'(M_1) \oplus \pi'(M_2)$. Since we know π', we can compute $\pi'(M_1)$ and thus get $\pi'(M_2)$. Finally, we invert this permutation to get M_2. We can thus decrypt any block once we know the single pair $(M_1, E_K(M_1))$. This attack reveals nothing about the key.

3. Block Cipher Statistics. *9 points per part; 45 total.* An obvious requirement of cryptographic systems is that if you have two different plaintexts $M_1 \neq M_2$ and encrypt them under the same key K, then the ciphertexts have to be different: $E_K(M_1) \neq E_K(M_2)$. However, if you encrypt the same plaintext M under two different keys $K_1 \neq K_2$, it is entirely possible that $E_{K_1}(M) = E_{K_2}(M)$. This was important in figuring out how many known plaintext-ciphertext pairs we would need to successfully mount a brute-force attack, or a meet-in-the-middle attack against double encryption.

In this problem you will study the statistics of such 'collisions'. Some pointers to relevant probability tools that you may have forgotten are given in each part. *In general,* use k to represent the number of bits in the key, m to denote the number bits in a block, and then use your general answers to obtain specific numbers for DES, AES, and BabyBlock.

(a). Fix a plaintext block M and a candidate ciphertext block M'. What is the probability that there is *no* key K such that $E(K, M) = M'$? HINT: Choosing one random key and encrypting M under it to see if you get M' is , or should be, the same as choosing a random block and seeing if you get M'. Then think of multiple independent trials of this experiment.

Solution for (a) and (b). If we fix K, the probability of hitting M' is 2^{-m}, so the probability of missing M' is $1 - 2^{-m}$. The probability of missing M' with with all 2^k keys is consequently

$$(1 - 2^{-m})^{2^k}.$$

For BabyBlock we have $2^m = 2^k = 2^{16} = 65536$.

(b) Use the result of *(a)* to estimate, for each M, the number of ciphertext blocks that cannot be gotten by encrypting M. Do this for each of our three example block ciphers. What does this tell you about the likelihood of colliding keys for a given plaintext M, at least in the case of AES and the Baby Block cipher? (HINT: It is very helpful to know that $(1 - 1/n)^n$ is close to e^{-1} for large n, and that more generally $1 - x$ is well approximated by e^{-x} for x close to 0. This problem might give a rather inconclusive result for DES unless you are really clever about it.)

Solution for (a) and (b). If we fix K, the probability of hitting M' is 2^{-m}, so the probability of missing M' is $1 - 2^{-m}$. The probability of missing M' with with all 2^k

keys is consequently

$$(1-2^{-m})^{2^k}.$$

For BabyBlock we have $2^m = 2^k = 2^{16} = 65536$. For AES it is $2^m = 2^k = 2^{128}$ and for DES $2^m = 2^{64}$ and $2^k = 2^{56}$.

We can use the approximation suggested in (b) to conclude that the probability of missing a block M' when encrypting M is about $1/e \approx 0.368$, so roughly 37% of the 65536 blocks cannot be reached by encrypting a single plaintext block M, and a similar proportion for the AES blocks.

For DES the probability is $(1-2^{-64})^{2^{56}} \approx e^{-2^{-8}}$, according to the above approximation, which is about 0.996. Thus about 99.6% of the 2^{64} blocks are not the decryption of a given plaintext M.

(c) This is an extension of parts (a) and (b) above. Estimate the number of ciphertext blocks M' such that *exactly one* key maps M to M', exactly two keys, *etc.* (HINT: This uses the Poisson distribution.)

Let $p = 2^{-m}$ be the probability that a given block M' is hit by a single 'shot'—here a shot is the encryption of a fixed plaintext block M by a randomly chosen key K. Let $n = 2^k$ be the number of shots, and $\lambda = np$. Then the probability of being hit by r shots is about

$$\frac{e^{-\lambda}\lambda^r}{r!}.$$

This is the Poisson distribution. In our examples we have $\lambda = 1$ for Baby Block and AES, and $\lambda = \frac{1}{256}$ for DES. For the first two, this gives probabilities for $0, 1, 2, \ldots,$ keys, of

$$\frac{1}{e}, \frac{1}{e}, \frac{1}{2e}, \frac{1}{6e}, \frac{1}{24e},$$

etc.

For example, the proportion of ciphertext blocks that M encrypts to under more than one key is

$$1 - 2/e \approx 0.264,$$

so with Baby Block this means there are about 17000 such ciphertexts.

For DES, the probabilities are

$$0.996, 0.00389, 7.6\times 10^{-6}, 9.895\times 10^{-9}.$$

The proportion of ciphertext blocks that M encrypts to under more than one key is

$$1 - 0.996 - 0.00389 \approx 7.6\times 10^{-6},$$

so the number of such ciphtertexts is quite huge, about

$$7.6\times 10^{-6}\times 2^{64} \approx 1.4\times 10^{14}.$$

(d) Parts (a)-(c) should tell you that if you pick a plaintext block at random, there are likely to be many pairs of distinct keys that encrypt it to the same ciphertext block.

Suppose I go looking for such a collision, testing keys at random until I discover a match. About how many keys will I have to test to have better than even odds of finding a colliding pair? (HINT: Do you know the problem about the chances of two people in a room having the same birthday?)

Solution. This *is* the birthday problem: We have 2^m different birthdays. We want to know how many people we have to assemble in a room before the probability of a shared birthday is more tha 0.5. We will go through a careful solution of this later in class, but a good rule of thumb is that the number of people is roughly the square root of the number of possible birthdays, and thus we need to test about $2^{m/2}$ keys. For BabyBlock, for example, this means we need to check approximately 256 keys; for DES about 2^{32}, around 4 billion.

(e) What we've been showing is that although the function

$$M \mapsto E_K(M)$$

is one-to-one for each fixed key K, the function

$$K \mapsto E_K(M)$$

for a fixed is block M is almost surely not, at least not if we want our keys to behave like random permutations of the set of plaintext blocks. This may come as a bit of a surprise after the one-time pad. Estimate (do this just for AES and BabyBlock) the probability that this map is one-to-one. (HINT: Stirling's formula for $n!$). The answer, as you might expect after all this, is 'very small'. But is it very small like one in a million, or very small like one in a googol, or like one in a googolplex?

Solution. We are in the setting where we have $N = 2^m$ blocks and keys, with $m = 16$ and $m = 128$. Ideally, the function

$$K \mapsto E_K(M)$$

for fixed M acts like a randomly chosen function from an N-element set to an N-element set. There are N^N such functions, so the probability that a randomly-chosen function is a permutation is

$$\frac{N!}{N^N}.$$

By Stirling's formula, $N!$ is about $\sqrt{2\pi N} \cdot N^N/e^N$, so the probability that a given such map is a permutation is

$$\frac{2\pi N}{e^N}.$$

Even for BabyBlock, with $N = 2^{16}$, this is already extremely small, less than one in a googol. For $n = 2^{128}$, we're in googolplex territory: the denominator is too big to write down, even for a computer.