YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

CPSC 467a: Cryptography and Computer Security — *Handout #15*

Felipe Saint-Jean

October 27, 2005

Solutions to Midterm Examination

Instructions:

This is a closed book examination. *Answer any 5 of the following 6 questions.* Write the numbers of the **five** questions that you want graded on the cover of your bluebook. All questions count equally. You have 75 minutes. Remember to write your name on your bluebook and to justify your answers. Good Luck!

Problem 1: Hill cipher

Eve captures Bob's Hill cipher machine, which uses a 2-by-2 matrix $M \bmod 26$. She tries a chosen plaintext attack. She finds that the plaintext ba encrypts to HC and the plaintext zz encrypts to GT. What is the matrix M?

Solution:

Using the standard letter-integer conversion $a = 0$, $b = 1$, $z = 25$, $H = 7$, $C = 2$, $G = 6$ and $T = 19$. In the Hill Cipher encryption is done by $C = P \cdot A$, so the system we need to solve is

$$\begin{pmatrix} H & C \end{pmatrix} = \begin{pmatrix} b & a \end{pmatrix} \begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{10} \end{pmatrix}$$

and

$$\begin{pmatrix} G & T \end{pmatrix} = \begin{pmatrix} z & z \end{pmatrix} \begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{10} \end{pmatrix}.$$

Writing both equations as a single system and converting the letters into numbers we get

$$\begin{pmatrix} 6 & 19 \\ 7 & 2 \end{pmatrix} = \begin{pmatrix} 25 & 25 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{10} \end{pmatrix}.$$

Solving

$$\begin{pmatrix} 25 & 25 \\ 1 & 0 \end{pmatrix}^{-1} \begin{pmatrix} 6 & 19 \\ 7 & 2 \end{pmatrix} = \begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{10} \end{pmatrix}.$$

Now

$$\begin{pmatrix} 25 & 25 \\ 1 & 0 \end{pmatrix}^{-1} = \frac{1}{-25} \begin{pmatrix} 0 & -25 \\ -1 & 25 \end{pmatrix} \equiv \begin{pmatrix} 0 & 1 \\ 25 & 25 \end{pmatrix}.$$

Finally

$$\begin{pmatrix} m_{00} & m_{01} \\ m_{10} & m_{10} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 25 & 25 \end{pmatrix} \begin{pmatrix} 6 & 19 \\ 7 & 2 \end{pmatrix} = \begin{pmatrix} 7 & 2 \\ (25 \cdot 6 + 25 \cdot 7) & (25 \cdot 19 + 25 \cdot 2) \end{pmatrix} \equiv \begin{pmatrix} 7 & 2 \\ 13 & 5 \end{pmatrix}.$$

Problem 2: Information-theoretic security

(a) Consider the cryptosystem with $\mathcal{M} = \{a, b\}$ and $\mathcal{C} = \mathcal{K} = \{0, 1, 2\}$. The encryption function is given by $E_k(m) = (m + k) \bmod 3$. Is this system information-theoretically secure? Explain.

(b) Suppose now $\mathcal{M} = \{a, b\}$, $\mathcal{C} = \{0, 1\}$, and $\mathcal{K} = \{0, 1, 2\}$. Does there exist an information-theoretically secure encryption function on these sets? Explain.

Solution

part a

It is information-theoretically secure. The pre-images of any element $\mathcal{M}$ is $\mathcal{C}$ with the same probability distribution as without information about $\mathcal{C}$. As a result the entropy doesn't change. Notice that seeing a ciphertext leaks information about the key but no information about the message. That is what information-theoretically secure is all about.

part b

It is NOT information-theoretically secure. In a similar argument as before the pre-images of an element in $\mathcal{M}$ is not $\mathcal{C}$ with the same probability distribution. That is because the size of the key space $\mathcal{K}$ is bigger so there have to be two keys that map each element in $\mathcal{M}$ to the same one in $\mathcal{C}$. The result is a change in the probability distribution; therefore the entropy changes.

Problem 3: Feistel network

Happy Hacker was asked to implement a Feistel network, but he couldn't quite remember how stage i worked, so he wrote down the equations:

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= f(L_i \oplus R_i, K_i). \end{aligned}$$

However, Happy was having trouble figuring how to decrypt messages encoded in this way.

(a) Show why Happy couldn't come up with a general decryption algorithm by exhibiting a particular function f_1 which makes it impossible to recover (L_i, R_i) from (L_{i+1}, R_{i+1}) when f_1 is used for f in Happy's scheme.

(b) Would have decryption been possible using your function f_1 of part (a) if Happy had gotten the Feistel network correct in the first place? Explain.

(c) Happy finally noticed that he could decrypt if he chose f to be

$$f_2(X, K) = X \oplus K.$$

Explain how to decrypt in this case.

(d) What can you say about the security of the system using Happy's function f_2 from part (c)?

Solution:

part a

Any non-invertible function will make L_i to be lost. For example $f(X,K) = K$ makes the cryptosystem non-decryptable.

part b

Yes, a correct Feistel network does not require inverting the function f so any deterministic function decrypts correctly.

part c

The encryption is

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= L_i \oplus R_i \oplus K_i. \end{aligned}$$

so the decryption is

$$\begin{aligned} R_i &= L_{i+1} \\ L_i &= R_{i+1} \oplus L_{i+1} \oplus K_i. \end{aligned}$$

To decrypt we need to run backwards from $i = n, \ldots, 1$, where n is the number of rounds.

part d

To build a cryptosystem out of a Feistel network several rounds are used (as in DES). Doing some rounds using the given round we get

$$\begin{aligned} L_1 &= R_0 \\ R_1 &= L_0 \oplus R_0 \oplus K_0 \\ L_2 &= L_0 \oplus R_0 \oplus K_0 \\ R_2 &= R_0 \oplus L_0 \oplus R_0 \oplus K_0 \oplus K_1 \\ L_3 &= R_0 \oplus L_0 \oplus R_0 \oplus K_0 \oplus K_1 \\ R_3 &= R_0 \oplus L_0 \oplus R_0 \oplus K_0 \oplus K_1 \oplus L_0 \oplus R_0 \oplus K_0 \oplus K_2 \\ &\vdots \end{aligned}$$

So after a few rounds we that that $L_i = R_0^\alpha \oplus L_0^\beta \oplus K_R$ and $R_i = R_0^\gamma \oplus L_0^\delta \oplus K_L$ where the exponent is over the operation $\oplus$. If the exponent is even then the element is zero. α, β, γ and δ depends only on the number of rounds and K_L and K_R are $\oplus$ of some K_i. As a result of this a known plaintext attack will reveal K_L and K_R, and that is all that is needed to decrypt. That means that Happy's cryptosystem will be weak against known plaintext attack.

Problem 4: Chaining modes

Let $E_k(m)$, $D_k(c)$ be a block cipher. Fischer Spiffy Mixer mode (FSM) encrypts a sequence of message blocks $m_1, m_2, \ldots$ by the sequence of ciphertext blocks $c_1, c_2, \ldots$ using the following method:

$$c_i = m_{i-1} \oplus E_k(m_i \oplus c_{i-1})$$

m_0 and c_0 are fixed (public) initialization vectors.

(a) Describe how to decrypt.

(b) Suppose ciphertext block c_3 is damaged in transit. Which plaintext blocks become undecipherable as a result? Explain.

Solution:

part a

XORing m_{i-1} to both sides of the encryption equation gives

$$c_i \oplus m_{i-1} = E_k(m_i \oplus c_{i-1}).$$

Applying the decryption function on both sides gives

$$D_k(c_i \oplus m_{i-1}) = m_i \oplus c_{i-1},$$

so

$$m_i = c_{i-1} \oplus D_k(c_i \oplus m_{i-1}).$$

part b

If c_i was damaged then m_i is damaged. If m_i is damaged m_{i+1} is damaged. From then on all messages are damaged.

Problem 5: RSA decryption exponent

Bob chooses an RSA modulus $n = 13 \times 7 = 91$.

(a) He wants an easily-remembered encryption exponent, so he wants to use either $e = 10$ (the number of decimal digits) or $e = 26$ (the size of the English alphabet). However, one of these will not work. Which one won't work and why?

(b) Since Bob didn't study for his crypto midterm, he couldn't answer part (a). To play it safe, he decided to stick to primes, so he choose $e = 17$. Find the corresponding decryption exponent d and show how you derived it.

Solution:

part a:

The condition is that $ed \equiv 1 \bmod \phi(n)$. For e to have an inverse we need that $\gcd(b, \phi(n)) = 1$. Since $\phi(91) = 72$ and both 10 and 26 are even we know that the *gcd* is at least 2 so none has inverse $\bmod 72$.

part b:

To find the inverse we need to use the Extended Euclid's Algorithm.

$$\begin{aligned} 72 &= 1 \cdot 72 + 0 \cdot 17 \\ 17 &= 0 \cdot 72 + 1 \cdot 17 \\ 4 &= 1 \cdot 72 - 4 \cdot 17 \\ 1 &= -4 \cdot 72 + 17 \cdot 17 \end{aligned}$$

Therefore $17 \cdot 17 \equiv 1 \bmod 72$ so $d = 17$.

Problem 6: An attack on RSA

Bob received an RSA-encoded message c from Alice. He decrypted it using the fast modular exponentiation algorithm described in class and reproduced here:

```
/* computes m^e mod n iteratively */
int modexp( int m, int e, int n)
{
  int r = 1;
  while ( e > 0 ) {
    if ( (e&1) == 1 ) r = r*m % n;
    e /= 2;
    m = m*m % n;
  }
  return r;
}
```

Nasty manages to break into Bob's machine and to get a snapshot of the stack frame of Bob's process while it was in the middle of decrypting c. In this way, Nasty learns the values of the variables `r`, `e`, `m`, `n` as they were at some instant in time during the execution of `modexp`.

(a) Explain why `modexp` is relevant to Bob's task of decrypting c.

(b) Explain how Nasty can use the values he has captured to decrypt c.

(c) Alice sends Bob another RSA-encoded message c'. Can Nasty also decrypt it with the information already at hand? Why or why not?

Solution:

part a

RSA decryption requires that Bob compute $c^d \bmod n$. The call `modexp`(c, d, n) does that.

part b

What Nasty does is just keep running the algorithm to finish it. He might not know exactly in what line of the program he got the variables from, but it is a small number, and he can try all possible starting points.

part c

In general he will not be able to decrypt a different message. Nasty did gain some bits of d that will reduce his search space in a brute force attack, but unless he was lucky and captured Bob's values near the beginning of the computation (when most of the bits of d were still available in `e`), he won't be able to decrypt arbitrary messages without a big effort.

(end of exam)