T-79.159 Cryptography and Data Security **Spring 2003**
Tutorial 1: Solutions
21.–22.1.2003

1. By the traditional definition, cryptography is the art of encryption, cryptanalysis is the art of breaking encryptions, and cryptology the study of both cryptography and cryptanalysis. A cryptogram is a secret message (i.e. a ciphertext). Cryptospore is a biological term completely unrelated to cryptology. Thus, the words should be connected as follows.

Cryptology:	Study of both using and breaking ciphers.
Cryptography:	Encrypting messages.
Cryptanalysis:	Breaking secret messages.
Cryptogram:	A secret message.
Cryptospore:	Cells of (unknown) ancient organisms.

2. For readability, we will write the ciphertext using upper case and the plaintext using lower case. Since the ciphertext is a dedication, the first word is probably 'to'. The most common letters in the ciphertext are I (4 occurrences) and G, K, O, Y (3 occurrences). The most common letters in English are e, t, a, o, i, n, s, h and r. Since we have guessed that the plaintext starts with 'to', we have the guess $S \sim t$ and $I \sim o$. Since e is the most common letter in English, we guess that e corresponds to one of $\{G, K, O, Y\}$. We arbitrarily guess $Y \sim e$. With these guesses, the message looks like

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to t e           o          o   o    e   e
```

The second word it probably 'the'. Since r is a common letter in English, we could guess that r corresponds to one of $\{G, K, O\}$. We arbitrarily guess $r \sim G$. This gives the message

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the           or         o r o    e  re
```

The fifth word is probably 'for', and since the text is a dedication, the seventh work could be 'our'. Thus, we guess $f \sim B$ and $u \sim O$. This gives

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the      u    for        our o    e ure
```

Since the eight word is preceeded by 'our', it is probably a noun in plural form. Thus, we guess that $s \sim K$. With this guess, it is clear that the last word is 'secure' (after all, it is a book on security). This gives

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the      u s  for        our  o s secure
```

The most common unknown ciphertext letters are N, E and W. It seems natural that one of these corresponds to a. On the other hand, the sixth word must be a verb. In the dedication, the authors probably thank somebody for doing something, so the verb probably end in 'ing'. This gives the guess $i \sim J$, $n \sim A$ and $g \sim W$. We are left with the guess that a is one of $\{N, E\}$. The guess $a \sim N$ gives

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the a   gu s  for    ing our  oas secure
```

whereas the guess $a \sim E$ gives

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the  a  gu s  for  a ing our  o s secure
```

The word ending with 'oas' looks strange, so let us stick with the guess $a \sim E$. Considering the subject of the book and its style, the third and forth words could be 'bad guys'. This would give the message

```
SI SPY NET WORK, BIG FEDJAW IOG LINK KYXOGY
to the bad guys  for  a ing our  obs secure
```

Now the message is already quite readable. We probably have $m \sim F$, $k \sim D$ $j \sim L$, and the plaintext is

To the bad guys, for making our jobs secure.

Note that we where very lucky in our guesses. In practice, we would have to try many more possibilities. Note also that we had some knowledge of what the message was about, which made the task much easier.

3. Let h be a hash function. When seeing the message $(M, h(M))$, the adversary can pick any message M', compute $h(M')$, and replace the message with $(M', h(M'))$. Since the hash function is correctly computed, the modification cannot be detected. Since hash functions does not depend on a key, everybody can compute them, and thus produce fake authentication tags.

   (This scheme does detect random modifications e.g. due to transmission errors, but there are much better error correction methods to achieve this.)

4. When A identifies itself to (say) B, B sends a random challenge r to A. A them computes $f(K_A, r)$, where f is some function, and sends the result to B. In order to be able to verify that the response is correctly computed,

B must know K_A. But then B can identify itself as A to C using K_A instead of K_B.

In order to use symmetric key methods to identify each other, each user must have two keys, A having keys K_{AB}, K_{AC}, B having keys K_{AB}, K_{BC}, and C having keys K_{AC}, K_{BC}. A can then use K_{AB} to identify itself to B and K_{AC} to identify itself to C.

5. Let h be a hash function. Since public-key operations are much slower (about 1000 times) than symmetric-key operations, it is common to sign the shorter $h(M)$ instead of the message M. The most important property of h is that it must be (strongly) collision-resistant. If one could find two messages x, y such that $x \neq y$ and $h(x) = h(y)$, a valid signature of x would also be a valid signature of y. Given the signed message $(x, \text{sign}_{K_s}(h(x)))$, the adversary could form the signed message $(y, \text{sign}_{K_s}(h(x)))$, which would be accepted as a valid signature, since $h(x) = h(y)$.

6. There was a small typo in the problem statement: R_b in the last step should of course be r_B.

   We will present an attack that allows the adversary Eve to identify herself as Alice to Bob (without knowing K_{AB}). We assume that Eve can open two connections to Bob at the same time (a very realistic assumption when Bob is a server and not a human being).

   In the first connection, E proceeds as follows.

   - Eve picks a random challenge $r_{E,1}$ and sends it to Bob.
   - Bob encrypts $r_{E,1}$ with K_{AB} and sends the result $y_{B,1} = E_{K_{AB}}(r_{E,1})$ to Eve.
   - Bob picks a random challenge $r_{B,1}$ and sends it to Eve.

   Eve now pauses the first connection, and opens a second connection to Bob. In this second connection, Eve proceeds as follows.

   - Eve uses $r_{E,2} = r_{B,1}$ instead of the random challenge and sends it to Bob.
   - Bob encrypts $r_{E,2}$ with K_{AB} and sends the result $y_{B,2} = E_{K_{AB}}(r_{E,2})$ to Eve.

   Eve now stops the second connection, and resumes the first connection. She proceeds as follows.

   - Eve sends $y_{E,1} = y_{B,2}$ to Bob.

   Since $y_{E,1} = E_{K_{AB}}(r_{E,2}) = E_{K_{AB}}(r_{B,1})$ is exactly the response Bob expects in the last step of the first connection, Bob accepts it and identifies Eve as Alice.

7. Let n denote the key length. The work load of legitimate users of the scheme is $L(n)$ for some linear function L. Since L is linear, $L(a + b) = L(a) + L(b)$ and thus $L(2n) = 2L(n)$. The best attack on the scheme is a brute-force attack trying all 2^n possible keys. On the average, the attacker must try half of the keys before the correct key is found. Thus, we assume

that the work load of the attacker is $A(n) = a(n) \cdot 2^{n-1}$, where $a(n)$ is the time is takes to process 1 key. If the computational resources doubles, the work load of legitimate users become $L'(n) = L(n)/2$ and the load of the adversary become $A'(n) = A(n)/2 = a(n)2^{n-2}$. If we would double the key length to $2n$, the work load of legitimate users and adversaries with the new computational resources become $L'(2n) = L(n)$ and $A'(2n) = a(2n)2^{2(n-1)} = \frac{a(2n)}{a(n)^2}A(n)^2$. In practice, this term is dominated by $A(n)$. Thus, the work load of legitimate users remains the same, but the work load of the adversaries grows quadratically.

In this sense, the legitimate users benefits more from the increase in computational resources.

8. The Fermat numbers $F(0) = 3$, $F(1) = 5$, $F(2) = 17$, $F(3) = 257$ and $F(4) = 65537$ are primes (as can be checked e.g. by trial division), but $F(5) = 4294967297 = 641 \cdot 6700417$ is composite. Thus, Fermat's conjecture is false.

   The following two Fermat numbers are also composite, since

$$\begin{aligned} F(6) &= 18446744073709551617 = 274177 \cdot 67280421310721 \ , \text{ and} \\ F(7) &= 340282366920938463463374607431768211457 \\ &= 5704689200685129054721 \cdot 59649589127497217 \ . \end{aligned}$$

   Standard computer algebra systems such as Maple and number theory packages like PARI can be used to factor these numbers. The Fermat number $F(5)$ can be factored by naive trial division, but remember that $F(5) > 2^{32}$ and thus cannot be represented as a 32-bit integer.