CS 415: Computer and Network Security

Fall 2007

Lab exercise: Vigenere, RSA, DES, and Authentication Protocols

Abstract: This lab is to give you some experience with cryptanalysis of historic ciphers, working with modern encryption algorithms, and understanding of authentication protocols.

1 Tools required for this lab:

A computer, the files listed within, and web access.

2 Background Reading

- OpenSSL can be downloaded from http://openssl.org, though commonly installed. RTFM.

3 Lab Exercises:

Include README files where appropriate to aid in grading this lab. Make a sub-directory for each section of the lab as appropriate.

3.1 Vigenere

Read the following:

http://www.trincoll.edu/depts/cpsc/cryptography/vigenere.html

Keyword:	**RELAT IONS**R ELATI ONSRE LATIO NSREL
Plaintext:	TOBEO RNOTT OBETH ATIST HEQUE STION
Ciphertext:	KSMEH ZBBLK SMEMP OGAJX SEJCS FLZSY

In 1863, a Prussian major named Kasiski proposed a method for breaking a Vigenere cipher that consisted of finding the length of the keyword and then dividing the message into that many simple substitution cryptograms. Frequency analysis could then be used to solve the resulting simple substitutions.

Kasiski's technique for finding the length of the keyword was based on measuring the distance between repeated bigrams in the ciphertext. Note that in the above cryptogram the plaintext bigram 'TO' occurs twice in the message at position 0 and 9 and in both cases it lines up perfectly with the first two letters of the keyword. Because of this it produces the same ciphertext bigram, 'KS.' The same can be said of plaintext 'BE' which occurs twice starting at positions 2 and 11, and also is encrypted with the same ciphertext bigram, 'ME.' In fact, any message encrypted with a Vigenere cipher will produce many such repeated bigrams. Although not every repeated bigram will be the result of the encryption of the same plaintext bigram, many will, and this provides the basis for breaking the cipher. By measuring and factoring the distances between recurring bigrams -- in this case the distance is 9 -- Kasiski was able to guess the length of the keyword. For this example,

Location	01234 56789 01234 56789 01234 56789
Keyword:	RELAT IONSR ELATI ONSRE LATIO NSREL
Plaintext:	TOBEO RNOTT OBETH ATIST HEQUE STION
Ciphertext:	KSMEH ZBBLK SMEMP OGAJX SEJCS FLZSY

the Kasiski method would create something like the following list:

Repeated Bigram	Location	Distance	Factors
KS	9	9	3, 9
SM	10	9	3, 9
ME	11	9	3, 9
...			

Factoring the distances between repeated bigrams is a way of identifying possible keyword lengths, with those factors that occur most frequently being the best candidates for the length of the keyword. Note that in this example since 3 is also a factor of 9 (and any of its multiples) both 3 and 9 would be reasonable candidates for keyword length. Although in this example we don't have a clear favorite, we've narrowed down the possibilities to a very small list. Note also that if a longer ciphertext were encrypted with the same keyword ('RELATIONS'), we would expect to find repeated bigrams at multiples of 9 -- i.e., 18, 27, 81, etc. These would also have 3 as a factor. Kasiski's important contribution is to note this phenomenon of repeated bigrams and propose a method -- factoring of distances -- to analyze it.

Once the length of the keyword is known, the ciphertext can be broken up into that many simple substitution cryptograms. That is, for a keyword of length 9, every 9-th letter in the ciphertext was encrypted with the same keyword letter. Given the structure of the Vigenere tableau, this is equivalent to using 9 distinct simple substitution ciphers, each of which was derived from 1 of the 26 possible Caesar shifts given in the tableau. The pure Kasiski method proceeds by analyzing these simple substitution cryptograms using frequency analysis and the other standard techniques. A variant of this method, proposed by the French cryptographer Kerckhoff, is based on discovering the keyword itself and then using it to decipher the cryptogram. In Kerckhoff's method, after the message has been separated into several columns, corresponding to the simple substitution cryptograms, one tallies the frequencies in each column and then uses frequency and logical analysis to construct the key. For example, suppose the most frequent letter in the first column is 'K'. We would hypothesize that 'K' corresponds to the English 'E'. If we consult the Vigenere tableau at this point, we can see that if English 'E' were enciphered into 'K' then row G of the table must have been the alphabet used for the first letter of the keyword. This implies that the first letter of the keyword is 'G'.

The problem with this "manual" approach is that for short messages there are often several good candidates for English 'E' in each column. This requires the testing of multiple hypotheses, which can get quite tedious and involved. Therefore we need a more sensitive test to discover the alphabet used by each letter of the keyword. Recalling that each row of the Vigenere tableau is one of the 26 Caesar shifts, we can use the chi-square test to determine which of the 26 possible shifts was used for each letter of the keyword. This modern day version of the Kerckhoff method turns out to be very effective.

Take a look at the following Java applet for analyzing Vigenere ciphers

http://math.ucsd.edu/~crypto/java/EARLYCIPHERS/Vigenere.html

Read the instructions on the web page above.

Retrieve the ciphertext from the course website. This original plaintext has been encrypted with the above applet.

Question 1: Using the applet, determine the key used to encrypt this text. The key consist of only alphabetic characters. Write down the key as your answer to this question.

Question 2: Print the first 10 lines of your successfully decrypted plaintext.

Question 3: Describe the technique implemented by the applet to recover the key? Is it using the Kasiski or Kerchoff methods? Explain why or why not.

3.2 RSA

OpenSSL is a full featured encryption toolkit that is used to implement secure services. Using openssl create a 2048-bit RSA key pair.

Question 4: Create a directory for this part of the lab. Store your private key as private.key and your public key as public.key. Create a text file input.txt that contains your full name. Encrypt the text file with your private key and save it as input.rsa. Turn in your public key and encrypted file, these will be used to verify.

Question 5:Using OpenSSL extract p, q, n, e,and d from your public-private key pair above. Save these out to a separate text file.

3.3 DES

For this part of assignment, you can use the regular edlab machines or your home computer.
Download the DES implementation (ftp://ftp.uu.net/pub/security/des/fast-des3.0.tar.Z) for lab2.
There is nothing special about this DES implementation other than it compiles with no effort on the edlab machines.

Tasks:
1. gunzip and Untar ("tar -xf") the file and then run the makefile ("make" on the command line).
2. Encrypt a short message.
3. Hand in your plaintext, cipher key, and cipher text.

Question 6: Print out the des() and desFunc() functions in desblock.c and with a pen mark up the code commenting on how it's following the DES algorithm.

Question 7: Search through the source code and identify where the program checks for weak DES keys. Include both the code to test for weak keys and the weak keys.

3.4 Authentication Protocols

Re-read Bishop Chapter 11 Section 11.3.4: He discusses a technique developed by Bellovin and Merrit called encrypted key exchange that defeats type 1 dictionary attacks again challenge-response authentication protocols. Create two subdirectories, one for Alice, one for Bob. Implement the encrypted key exchange protocol using the OpenSSL command line tools for encryption, and copying files between each directory as transport. Assume as in Bishop that Alice and Bob share an initial symmetric key a priori. Use OpenSSL to generate random data whenever necessary.
Remember that this is an exercise to see how authentication protocols operate. This is not a protocol that would be used in the real world.

Hint. Choose an RSA key length sufficient to encrypt the DES key output (2048 bit key worked for me…)

Question 8: Write up the steps you took for each step of the protocol. Include the command line arguments for each call the OpenSSL, as well as which files you copied where.

Question 9: How would you modify the encrypted key exchange protocol if Alice and Bob did not share an initial symmetric key?

3.5 Evaluation

Evaluation

Question 10: How hard was this lab? Was it fair? How would you change it to improve it?

3.6 Bonus (worth an extra 5 points, not mandatory)

Question 11: What noted cryptographer was a signatory to the plaintext document used in section 3.1? Describe briefly what he in known for in the context of cryptography.