SCI 199 L0161 Cryptography Exercise, 2005-2006

Public-Key Cryptography

This is a class exercise to explore some of the operation of ‘‘public-key cryptography’’, which is a class of cryptosystems in which it is possible for two parties to establish secure communications *without* a prior secure communication.

In a conventional cryptosystem, the parties exchange a ‘‘key’’ in advance. This traditionally happens in a meeting in person. At a subsequent time, the parties can communicate securely even though they are separated and their conversation may be overheard, because the sender encrypts the communication with the secret key, and the recipient decrypts the communication with the secret key.

In a ‘‘public-key’’ cryptosystem, the information you need to send a secure communication to the other party can be made publicly available. Everyone has a ‘‘key pair’’, which consists of a ‘‘private key’’ and a ‘‘public key’’. There is a mathematical relationship between these keys, such that if person ‘A’ (which stands for Alice) encrypts a message using her private key and person ‘B’ (Bob)’s public key, then Bob can decrypt it only using Alice’s public key and Bob’s private key. Since no one other than Bob knows Bob’s private key (this might be a rash assumption), only Bob can decrypt the message (hopefully). Since his decryption of the message involves Alice’s *public* key, the fact that the message was decrypted successfully proves that it was encoded by Alice (or by someone else who knows Alice’s private key).

Public keys for such a system can be posted publicly, e.g. on someone’s web page so that you can send them secure e-mail.

1. Generate a key-pair

Part 1 of this class exercise is to generate a key-pair. Generally speaking, you run a computer program to do this—to generate some numbers which are somewhat random, but have the correct relationship between the two numbers in the pair. For this SCI 199 exercise, the software you need is being made available on the web. In fact, the key-pairs have all already been generated, one key-pair for each student. To find the key-pair I’ve generated for you, look at the grades page on the course web page and select ‘‘my key-pair for cryptography exercise’’.

There are three numbers generated to form a key-pair: a ‘‘modulus’’, an encryption exponent, and a decryption exponent. Your ‘‘public key’’ is the encryption exponent and the modulus. Your ‘‘private key’’ is the decryption exponent and the modulus.

2. Disseminate the key-pair

If another student is to be able to send you a message, they need to know your public key and you need to know theirs.

Publish your public key by sending e-mail to me (ajr). I will post them all on the web at http://www.dgp.toronto.edu/~ajr/199/ex/prv/publickeys.html

Since it will contain all students’ real names, this web page will be password-protected for privacy from the rest of the world; the password will be announced in class, or please ask by e-mail if you forget or miss it.

3. Choose your recipient and message

In the tutorial on October 6th, all students’ names will be written on index cards and you’ll receive a random one to send a message to.

To make the calculations less unwieldy, the message will be restricted to ten ASCII characters. So you can only say a couple of words, or it could be shorter than ten characters. (Please don’t send inappropriate messages.)

4. Encode your message

I’ve written a program to do the encoding as well, but you should try to follow the numerical calculations

(over)

to at least some extent. You can type the relevant keys and the message into a form at http://course.dgp.toronto.edu/cgi-bin/encode and it will show you the calculation.

First of all, encode all of the characters in the ASCII character set in binary. At seven bits per character, this gives you up to 70 bits of message. Consider this to be one big 70-digit number, ‘‘M’’. (Or however many digits it works out to, if your message is shorter than 10 characters.)

(The numerical calculation which ensues will be discussed on a web page with some rationale; I realize that some students will follow more of it than others. Basically, you raise this message value to an exponent which is the recipient's public key times your private key, discarding multiples of the modulus. The recipient raises the cryptotext to an exponent which is the sender's public key times the recipient's private key, again discarding multiples of the modulus. The relationship between the public keys, private keys, and moduluses is such that the original message is obtained. The discarding of multiples of the modulus means that you can't reverse the encryption by simply dividing. The use of large numbers (we're going to use 71-bit primes just because it has to be bigger than the message, but normally people use more like 1000 bits) makes it impossible simply to try all possibilities.)

5. Attempt to transmit your message to the desired recipient

E-mail the result to me, ajr. State whom your message is from and whom it is to, and I'll post them all on a course web page. Thus for the purpose of this example, not only will your desired recipient be able to see the message, so will everyone else in the class... but they'll just get the encrypted message. Only the desired recipient will be able to interpret the meaning of the message.

Similar to in step 2, this web page will be password-protected for privacy from the rest of the world; the password (which is the same as in step 2) will be announced in class, or please ask by e-mail if you forget or miss it.

6. Decode the message which someone else sent to you

Look through the list of available messages to see which one is for you. As in step 4, I've written a program to do the decoding for you, but you should try to follow the numerical calculations to at least some extent. You can type the relevant keys and the message into the form at http://course.dgp.toronto.edu/cgi-bin/decode and it will show you the calculation.

Discussion questions

What exactly does the decryption prove, in terms of authentication? Who else knows Bob's private key? In how many places can you spoil this system by eavesdropping?

Suppose that no one knows my private key; what are the ways in which can someone intercept your secret message to me? What is the ‘‘man in the middle’’ attack?

How does your web browser know the public key for your bank so as to provide an ‘‘SSL’’ (‘‘https’’) connection to it? How do we manage public keys on a large scale?

What would we do if one student sent another student an extremely inappropriate message? What would happen when the recipient complained? How could this turn into a successful academic offence prosecution despite the encryption?

Actually, I (ajr) can decrypt all of the messages anyway. How? Hint: There is no defect in the mathematical algorithm, which is indeed secure by modern standards. Also, I'm not relying on the short key length.