SYMMETRIC AND PUBLIC KEY ENCRYPTION LESSON

Mark Emry, McNeil High School, Round Rock ISD, Austin, TX

This material is based upon work supported by
the **National Science Foundation** under Grant No.1548315.

SYMMETRIC AND PUBLIC KEY CRYPTOGRAPHY

MULTI-DAY PROGRAMMING EXERCISE: DIFFIE-HELLMAN KEY EXCHANGE METHODOLOGY

During this exercise students write a program which models a simplified version of the Diffie-Hellman key exchange.

PURPOSE

The three purposes of this exercise are

1. for students to develop a deeper and more connected understanding of modern cryptographic key exchange methods,
2. for students to become more practiced programmers, and
3. to connect student learning between simple Caesar Ciphers and the substantively more complex Public Key Encryption methodologies.

Figure 1 Photograph: Rod Searcey and Linda A. Cicero Copyright Associated Press. All Rights Reserved.

ASSIGNMENT

Model the Diffie-Hellman key exchange process. The starter code will include Global Constants which specify the Shared Prime (PUBLICPRIME) and Shared Generator (PUBLICGENERATOR) public keys. Two additional Global Constants will specify the private key of the first participant (ALICESECRET) and the second participant (BOBSECRET).

Students must successfully calculate **alpha** (i.e., Alice's public key which she calculates and sends publicly to Bob) and **beta** (i.e., Bob's public key which he calculates and sends publicly to Alice).

Using **beta** and Alice's private key (**ALICESECRET**) students must then successfully calculate Alice's shared secret (**aliceSharedSecret**). Additionally, using alpha and Bob's private key (BOBSECRET), students must successfully calculate Bob's shared secret **(bobSharedSecret**). **aliceSharedSecret** and **bobSharedSecret** will match if students have successfully completed this assignment. alpha and beta, on the other hand, will neither match nor provide Eve with any insight into Bob or Alice's shared secret.

A user-defined function, OutputResults, has been supplied as part of the starter code so that students can display the following variables in uniform fashion:

- public prime number (**PUBLICPRIME**)
- public generator number (**PUBLICGENERATOR**)
- Alice's secret number (**ALICESECRET**)
- Bob's secret number (**BOBSECRET**)
- Alice's calculated result which she shares publicly with Bob (**alpha**)
- Bob's calculated result which he shares publicly with Alice (**beta**)
- Alice's calculated shared secret number (**aliceSharedSecret**)
- Bob's calculated shared secret number (**bobSharedSecret**)

EXTRA CREDIT

The algorithm which students will create here results in a shared secret without any encryption. Students seeking an extra challenge may elect to use the shared secret number (ie., **aliceSharedSecret** or **bobSharedSecret**) created by their Diffie-Hellman Key Exchange Algorithms to encrypt and decrypt following plaintext message: IF YOU DONT EAT YOUR MEAT YOU WONT GET ANY PUDDING

Students should reuse the encryption and decryption algorithms they created as part of the multi-day programming exercise in the previous Caesar Cipher CCL. In this case, the shared secret number created by their Diffie-Hellman Key Exchange algorithm (e.g., nine from the example on the presentation) should be used as the Caesar Cipher **shift** for both encryption and decryption purposes. Students should feel free to reuse code from their programming exercise as part of the Caesar Cipher CCL.

In this case, Eve will see all of the following:

- public prime number (**PUBLICPRIME**)
- public generator number (**PUBLICGENERATOR**)
- Alice's calculated result which she shares publicly with Bob (**alpha**)
- Bob's calculated result which he shares publicly with Alice (**beta**)

It should be noted that Caesar Ciphers are not considered cryptographically sound given their susceptibility to frequency analysis. For purposes of this exercise, however, more cryptographically sound encryption algorithms (e.g, DES) are beyond the scope of this exercise. The purpose here is for students to understand that the key exchange part of the Diffie-Hellman Key Exchange Algorithm is different from **using** the resulting shared secret (e.g., nine from the example in the presentation) to encrypt and decrypt messages.

CALCULATIONS

alpha = PUBLICGENERATOR ^ ALICESECRET MOD PUBLICPRIME

beta = PUBLICGENERATOR ^ BOBSECRET MOD PUBLICPRIME

aliceSharedSecret = beta ^ ALICESECRET MOD PUBICPRIME

bobSharedSecret = alpha ^ BOBSECRET MOD PUBICPRIME