Exercise Sheet 3

Send your solutions to ics-ex@goesec.de

Deadline: Wednesday, November 25th, 11:59 a.m. CET

Self-evaluation  The script to check the format of your submission can be downloaded from the course web page: http://files.goesec.de/2015-ics/guidelines.pdf

For practical tasks we again provide unit tests as ex3.py. Test input and output are defined in ex3_testdata.py. For evaluating the exercises we will use a similar but different configuration that contains alternative inputs.

Please, do not forget to mention your name on each page of your report.

RSA

1. (3 points) Messages corresponding to the numbers 0, 1 and \( n - 1 \) have a special property when encrypted using the RSA algorithm. What is this property and proof that it holds true?

2. Alice uses the RSA algorithm for encrypting a message \( m \). Her public key (exponent) is 211 with modulus 67063.

   (a) (7 points) Implement a tool crack_rsa.py to automatically reveal the plaintext from a ciphertext \( c = 19307 \).

   usage: crack_rsa.py [-h] -e INT -n INT --ciphertext INT

   With \(-e\) specifying the exponent, \(-n\) the modulus, and \(--ciphertext\) the ciphertext to break. All values are specified as integers to simplify the task.

   The result (the revealed plaintext message) should be written to standard output (stdout) with no additional comments.

   (b) (1 point) Why do you succeed for the given example?
Diffie-Hellman

3. Three-party extension
   (a) (4 points) Modify the original Diffie-Hellman key exchange to support three
       parties. Use as few steps as possible for negotiating the key.
   (b) (2 points) Compare your solution with an RSA key exchange for three parties.

4. Alice and Bob use the Diffie-Hellman key exchange to negotiate a common shared
   key. They agree on using generator/base 10 with modulus 1783.
   An attack might eavesdrop the negotiation and, for instance, see that Alice sends
   929 and Bob sends 626 over the public transport.
   (a) (7 points) Implement a tool crack_dh.py to automatically derive the secret
       shared key from the publically transferred integers, which are exchanged by Al-
       ice and Bob.

       usage: crack_dh.py [-h] -g INT -n INT --alice INT --bob INT
       With -g specifying the generator, -n the modulus, and --alice and --bob the
       publically transferred integers.

       The derived key should be written to standard output (stdout) with no addi-
       tional comments.

   (b) (1 point) Why do you succeed for the given example?