Coursework for CSC3621 Cryptography

This is the third and final of the three blocks of coursework for CSC3621 Cryptography. The coursework consists of two exercises with 20 marks each. We have reduced the workload to two exercises as the coursework was released later than anticipated.

There are 10 marks available for using GNU Privacy Guard to submit your exercise results following the instructions below. Please note that we require you **to submit a plaintext version of your coursework in any case** to ensure that it can be marked even if the encryption went wrong.

Submission instruction

Submit the exercise results together as a zipped archive. If you do the GnuPG exercise, then submit two versions of your coursework: one in plaintext (which will be the authoritative for the marking) and one encrypted and signed as armored GnuPG file.

Deadline: Friday, 17:00, 12 Dec, 2013

Exercise III-1 (20 Marks)

Aim: To understand a RSA Encryption and how it can be attacked.

Background information: The material in this coursework was covered in **Lecture 13/4**, **RSA**. The coursework draws upon the Java class **BigInteger** to do modular computations.

Instruction: Breaking Textbook RSA

Question 1

Consider the following scenario and analyze what can go wrong in using textbook RSA.

A system administrator Alice sets up textbook RSA as protection mechanism for user passwords, providing each user with an RSA key pair as part of their initial rollout: sk=(N_{User}, d_{User}); pk=(N_{User}, 3). Thus, each user gets a separate RSA modulus N_{User} and decryption exponent d_{User}, whereas the encryption exponent is 3 for all users. The administrator distributes an 8-character login password pwd_{Bob} to a user Bob as follows:

$$c = (pwd_{Bob})^3 \ (\textbf{mod}\ N_{Bob})$$

Write a short paragraph analysing what can go wrong in this.
Comment what could go wrong if the administrator decided to use the same RSA modulus for the entire organization, only changing the public/private exponent pair per user.

Question 2

Now it's your turn to break RSA. We have a scenario as described in Question 1, that is, you (acting as eavesdropper) have received a ciphertext of which you know it is encrypted with e=3 and you know that the underlying message is eight characters long in an all-capital letter message space. Decrypt the ciphertext.

Public RSA key:

N =
23095675100376460353980581297675223373026833410647478222648288977449481620360427

e = 3

c = 67447252662059390380049763724240018791675318590 9

We use a very simple encoding of upper-case characters into numbers: Lookup the decimal ASCII-code of the character, and concatenate the codes. For instance, “GREAT” is encoded as follows:

G = 71, R = 82, E = 69, A = 65, T = 84. You obtain the codes 71 82 69 65 84
➔ The encoding of “GREAT” is 7182696584. It’s reversed by looking up the ASCII codes for every two-digit pair. Please submit your program to break this textbook RSA and the decrypted password.

Question 3

Explain how textbook RSA can be enhanced to render this and other attacks impossible and make it a CPA-secure encryption scheme. Name three major security benefits of the method. Submit one brief paragraph on the approach.

Submission Guideline

Write the answers to the Questions 1 & 3 in a short report (**text or Word file**). Include the **Java source code** for Questions 2 in the submission plus a **text file** with result of the decrypted password decoded as characters.

The submission is on NESS.

Marking:

20 marks in total	
Question 1:	10 marks
Question 2:	5 marks
Question 3:	5 marks

Exercise III-2 GnuPG Exercise (20 Marks)

Aim: Learn to use Gnu Privacy Guard (GnuPG) as means to secure e-mail conversations and file transfers with strong digital signatures and encryption.

Background information: Gnu Privacy Guard is an open source implementation of the Pretty Good Privacy (PGP) protocols. It supports public-key encryption and digital signatures, yet also comes with symmetric primitives such as hash functions (e.g., SHA1 or SHA25) or ciphers (e.g., AES).

The designated version for this exercise is: GnuPG 2.0.22 / GPG4Win Vanilla 2.2.1

Manual: http://www.gnupg.org/documentation/manuals/gnupg/

Installation:

The preferable approach for this exercise is to install your own version of GnuPG.

Downloads: http://www.gnupg.org/download/index.html

Windows Installers: GPG4Win - http://gpg4win.org/

Running GnuPG without Installation:

In some system configurations (e.g., the university windows setup), the execution of the GPG4Win installers requires administrator privileges, even if the intention is to install the tool into a user directory.

To make GnuPG accessible to all students, even if no own computer is available, we provided a portable version of GnuPG (GPG4Win Vanilla 2.2.1) which can be run without installation and be put on e.g. a USB drive. The portable installation is available on Blackboard.

Remark: Our quality assurance and tests of the software showed that GnuPG as portable edition is still is early development. In particular, we've had experiences with glitches on systems that had another installation of GnuPG, notably ones that broke the key generation/gpg-agent. To mitigate these side effects, we've created a Portable GnuPG with a batch file `local.bat` which creates a `cmd.exe` shell that enforces that GnuPG only considers the local directory. This edition has been tested on multiple operating systems and configurations and allows completing all tasks for the exercise.

1. Call `local.bat`
2. Work with `gpg2.exe` as main tool (not `gpg.exe` itself)
3. The call `gpg2.exe` --help provides a list with parameters.

Instruction: Putting theory into practice [10 Marks]

This part of the exercise asks you to write a few paragraphs each to establish how the parts GnuPG relate to topics covered in the lecture, for each of the questions below dedicate one paragraph. In total you should write at most **1 side A4**. The questions relate to the steps you take in the second part of this coursework and are mentioned there again to give you an opportunity to reflect as you experiment with the tool.

Question 1: Integrity Verification

How can you verify that the GnuPG installation to get is authentic and with integrity? How can you ensure that the public key of a recipient is indeed authentic? What means does GnuPG provide? Why is it important to verify both? How can you bootstrap an installation?

Question 2: Encryption of Large Files

Your coursework will likely be an archive, possibly larger than 3072 bits. Find out and explain how GnuPG actually encrypts larger files. How does encryption work if you encrypt towards multiple participants? How do you need to encrypt if you want to be able to decrypt the file later on?

Instruction: Sign and Encrypt your Coursework Submission [10 Marks]

Note: We require you to submit a second version of your coursework in plaintext to have a safety net, if something goes wrong with encryption or decryption.

You should not do this in practice for a real-world application!

Step 1: Verify the Integrity of your GnuPG Installation.

Determine two ways on how to verify that your GnuPG version indeed comes with integrity (i.e., has not been manipulated by an adversary). Reflect on how this constitutes a challenge in practice.

Step 2: Generate your own GPG key pair.

Create your own key pair to receive encrypted messages and to sign messages.

Parameters: RSA/RSA; 3072 bits; Use your real name and university e-mail address as identity.

Step 3: Export an armored version of your public key (see Appendix A).

Step 4: Import the public key of our demonstrators (see Appendix A).

How can you be convinced that this public key is actually integer? What steps can you take to verify that you got the right key?

Step 5: Sign and encrypt your coursework archive to the demonstrator key.

Perform a digital signature with the key associated with your university identity.

Encrypt the message to the demonstrator key with identifier 7F89AFBE. Make sure that you output an ascii armored ciphertext.

Submission Guideline

Submit an **armored** ciphertext to the demonstrator key, signed with your identity. Provide the armored version of your exported public key. Do this along with a plaintext copy of your coursework.

The submission is on NESS.

Marking:

20 marks in total
Question 1: Integrity Verification: 5 marks
Question 2: Encryption of Large Files: 5 marks
Armored Encryption: 5 marks
Signed with own SK: 5 marks

Appendix A: Demonstrator Public Key

Key identifier: 7F89AFBE

ASCII armored format.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2.0.22 (MingW32)
mQGNBFKWOWMBDAD5KFyBLstRbBQOtg3lzrP0XxfHRy4Hl2wOc6rRsFkfG8M8BeGn
vvzZ5vNhDsgOwpwDX0RI8MiHsSCoUIXcF+qVkpxSmp8UZjzsP8QjnRGWZtxWlClE
/HXuUUknG0Pty5eIgOZVlHrnqjRXchgo19g5mQyuphK218SMMo027IyvVwvewGz+
w5CycT8wZDfMQNdnrOTgK9qNm7OjfSyrCBxsWRTjWi+mR+vxCxf2yqo2xHeWeoYB
SgI/0axXlvnQGWf0es6eELxbHJ/85cDpM+/3DMJ0tn9JpFQwh674S3Am5xydX7/L
XXpDlUP8MPpACEDbuKkrP+4h5SKQxntEoFDeixxxmfosjjUIxrxLVG/QVVX8xr2k
ZlnXdFoZLK3nRUhfcnvP74AghJEjIP7lP5kch3p/ZghSE1JObjmbv3ke+ON404Qm
GCuUAc5tq1DnJZ9vd/kYx4pu9dpYSevlKbESFOTSG7W00HCzM/VK0BE5PTXJcTYE
NEnk/d1QiqJiY7cAEQEAAbRuQ1NDMzYyMSBEZW1vbnN0cmF0aW9uIEtleSAoTk8g
TEVHQUwgUkVMRVZBTkNZLiBTaGFyZWQga2V5IGZvciBkZW1vbnN0cmF0aW9uIHB1
cnBvc2VzIG9ubHkpIDxjc2MzNjIxQG5jbC5hYy51az6JAb8EEwECACkFAlKWOWMC
GwMFCRLMAwAHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIXgAAKCRCKJSMMf4mvvtSr
DACPt211giBT67AxLRHyxQUQFz49zE5dykLSflTIjLWvfIBMLZ9vZXbPmPBQdIXW
EoL2WEUSucXrFeYOKUU7vojqZ0tMKXEGBLCUmr3RQu3pBF2Ay2f/25JcX+sFPomJ
WC18S5ionXql941X2qFnEVo2HXjpuZbXYkRAH2CQahD43IAFqhba9wVvYTzRYkWc
CoTScfgGW6N4HIClP/8KSACgW4tvYGTTAbBpHS5N7SoPna0KVCjcRJfjxlnE2TuW
tmnsKwcYKmJpskS0HLQ8+OjrLyH248taFFEvqr51JdQv/akmvKDo7J0cZ8aZCoxS
PytTqFze5A7OevMcB6ccxIKwEjEZpprgjCdG5PYEaUPPpcFNJmyjSBY1QPtr1D30
++63Tc4eLTkWO6ZCUhOt/RZFG+4wtY97Gq4VLp4UgC7LYLqvDu1HkvgH1a9mDQuH
Xtkn4MnioxSBN0MIxzCbosWiJG5SENv3env32Y46h4UV8f3DRlfm/GUhoHiCRdGY
d/C5AY0EUpY5YwEMANdkqn3U/7oLdeYjqru47hzgbTRZ5ot+cFGqAjXmQfPXWojm
bRSKITMaSQmaRZkRv5hGcKSTmPjnOtAoRDTzhSLHARC+NTHAR6+j50HGGDG57ELn
L0zgIOBChyabUfJtqrzSqOhbs87FCfwKYyOvMPI2mCbvQIque1lzrFOdy2KPoeCD
5viIlhncP0IqMcXr86YxYFdh4pgY2RA3qvDysTPfw/fpSrtDRU9Aq8gKdr0Xj1W6
EowDhJ/cbNRjzCgIIc3vCMsa8OVbKxn3YhgLHvCffCzFiWO44lFaoSgCmSG655rI
azJEF4RpyZtIbQbCYCyPNDCAOjjJtuOJt2pfHT8rKwJph2E7jCKs0VxJitGXGFGi
InwpwyXUHyeIcJgTJv0gTISBQMK5WAOp+NfmyzGgRsfeUbtqCItBa9GgsD58D/Yn
aYYWj6Ln71vEkiqz4mSUeG9TuwPwzxkWhJNzTa0XnrPVgr1nx209hcTz496qv6h5
4P/s98P598jR3DKBVwARAQABiQGlBBgBAgAPBQJSljljAhsMBQkSzAMAAAoJEIol
Iwx/ia++wbMMAOD5Ds7oiMdlIB/Kwzf9tjn5WeyCrzi7qcCYj9cbwS6zFcYIcwTi
UsCOXvfhc0XqhX3gzyNh44soGXpEq6ADL2itT9ldFi8lzUoO2UrHpx5qu+pzbp0u
gDDAdHuj/YYpmmfvtLhGYbaFAC9Qs/iR4YKnqrzyQWRK17xEPgwWWqAwtowNfeT8
L5PzPal2Jp4TzdDbpLQ3/QvIYvdebp7t9IbxNm5gJtL2ByF2njvhkCZjshWR+CVZ
genLdVQYbIHOlESIQeJtcQdMILm67scB27VVo83gNQryrIL3lZBP0qm0+3C6R/Wb
NQZwR77TiZor3nKaqRI6yQcGZQrbsMXxqEvB0Vmyu24BOp9lEVGRVVJyTaJ3yLcL
iyGcHMyd92KTvTI7HxljpCsy5YSvhOGZRoi9RAqaGTAP+LT0Cel9QRcfX9KyqqOK
zf59LLYLUHOlPPnxq5Zi8aJaHkNK7k+om1UEtjkZ11zFxITWFKARYhJaUoLxofgj
1FjhGdCuU/R0Xw==
=reav
-----END PGP PUBLIC KEY BLOCK-----
```