University College London
Department of Computer Science

Cryptanalysis Exercises Lab 00

P. Spacek, N. Courtois

January 23, 2019

peter.spacek@stuba.sk
Version 1.0

1. Getting Acquainted with SAGE

1.1. How to run a LOCAL version of SAGE?

SAGEMath is a free open-source maths software package based on Python. We will be using SAGE for most of the lab sessions in this course.

- **Method 1 for Windows and Mac:**

The best method for large projects.

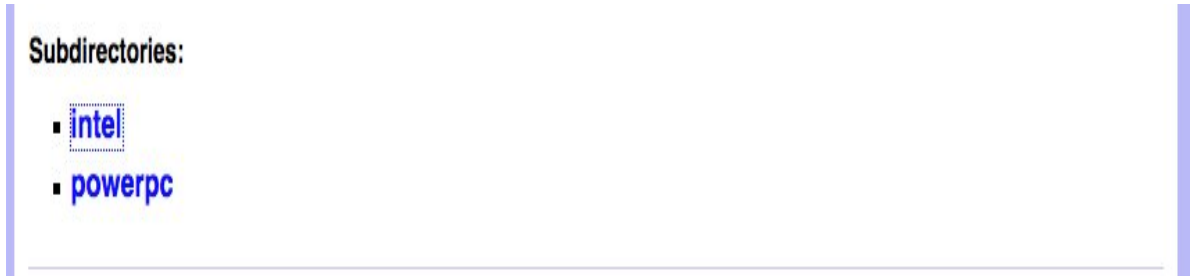
Download according to your OS:

1. Windows x64 - Native installer You need to select a mirror for downloading.

   HOW to use this?
   A shortcut called "SageMath 8.0" (plus two others that you can ignore) is created on your desktop. Double click, choose a password if requested and then type "notebook()". Wait for 5 seconds, a browser will open.
   Then click on "New Worksheet or on an existing worksheet".

Give it a name like "GA18_exo_1". Do not forget to press the button "Save" on the top right side, or "Save&quit". Then close the browser or browser tab, then go back to the terminal and type Ctrl+C and type "exit".

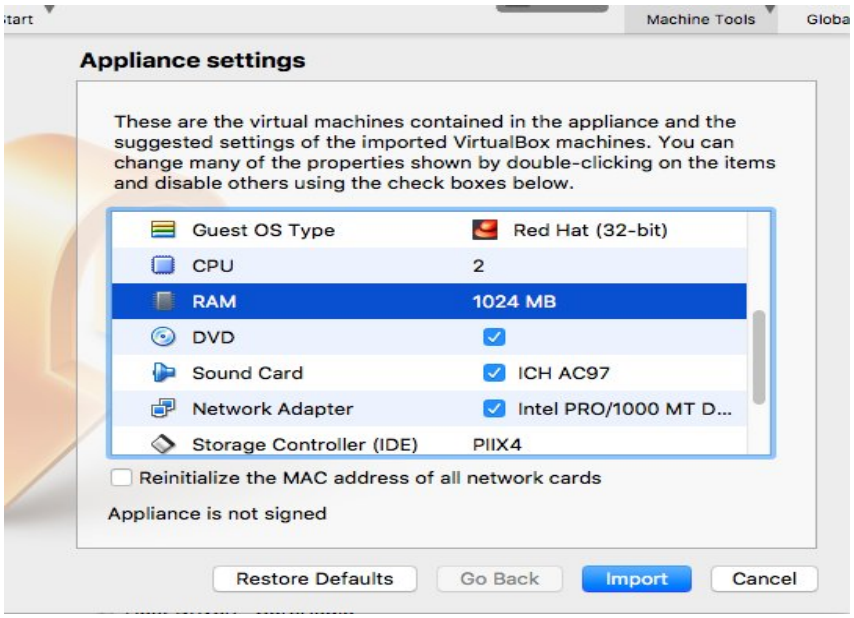
2. For MAC
   You need to pay attention to choose a correct version (a type of CPU: Intel vs PowerPC, version 8.0 and it needs to be app version)



Install it (move it to applications) and double click on the Sage icon (may need to)

- **Method 2 -Install VirtualBox:**
  1. Download links:
     For Windows PC or old builds
     For Mac OSX
     For Linux
  2. Obligatory extension to by run AFTER the installation, download here
  3. Install SAGE machine/apppliance.
     (a) Method 1: one single OVA file. import an OVA file: for example this file, 3 Gigabytes [highly compressed].

In Virtual Box go to "Import appliance". When importing it is very useful to increase the RAM and the number of CPUs and also to reset MAC address.



more about this, in principle not needed.

(b) Method 2: This method is better (for example) if you want to MOVE it from one machine to another or CLONE it with existing SAGE worksheets.

Copy 2 files from another PC after installation. One is machine definition file, on my PC called "Sage-6.9.vbox" and the other one is the hard drive file, on my PC called sage-6.9-disk1.vmdk, (could also be .vdi, there are several options). Then you have got to "Add" menu option inside the VirtualBox software and add the .vbox file and the check settings/Storage if it points to the right hard drive, or it need also added manually wrt to the current location on your PC.

More detailed instructions (not needed) for example if you wanted to do some advanced modifications, change the Linux keyboard layout, make it remotely accessible etc.

1.2. Using SAGE Math Cloud

SAGE can be accessed online via the SAGE Math Cloud.

1. First, click the link above and enter your details to register for a free account.
2. Once you have logged in successfully, click 'Projects' in the top

left-hand corner.

3. Click 'New Project'. Add a title (e.g. 'Cryptanalysis') and a description (e.g. 'Code for COMPGA18') and click 'Create Project'.
4. Click on the project you have just created.
5. Add a new file by clicking 'New'. Choose a name for your file (e.g. 'Lab Session 1'), and select 'SageMath Worksheet' as the type.
6. Now you are ready to write code! Type `3+4` and click run.
7. You can write many separate programs in the same window. You can also leave complex programs to run while you log out and do something else.

SAGE is built from Python. Most Python commands will work fine in SAGE, and details of more commands including number theory and algebra can be found here.

You can also install SAGE on your own computer by downloading here and using the installation guide here.

1.3. 50+ Basics of SAGE

Run these commands in SAGE. You can execute them with combination of "Shift" + "Enter" keys.

1. (3 + 4)%5
2. 3 + 4%5
3. 7 * 8%11
4. Zr=Zmod(5)
5. Zr
6. Zr(3+4)
7. Zr(3)+Zr(4)
8. inv3=Zr(3)^-1
9. inv3*3
10. ZZ
11. ZZ.cardinality()

12. G11=GF(11,'a') (older syntax)
13. G11
14. G11=FiniteField(11,'a')
15. G11.cardinality()
16. a**23 (does not work see below)
17. 'a'**23
18. G11.modulus()
19. G11.is_field() from sage.rings.finite_rings.finite_field_constructor import is_PrimeFiniteField; is_PrimeFiniteField(G11)
20. G11.base_ring()
21. G11.base_ring() is G11
22. G11.variable_name()
23. G11.<a>=GF(11); a**23;
24. a (very bizarre but correct!)
25. parent(a)

26. a.multiplicative_order()
27. F8.<b>=GF(8);
28. b**3;b^3; b**4;b**5;b**6;b**7
29. b.multiplicative_order() is b a primitive element? see slide 16 in here
30. F = Integers(5);F(3)+4:
31. list(F)
32. list(F8)
33. Integers(5) is Zmod(5)
34. IntegerModRing(5)
35. GF(9)(4)
36. k16=FiniteField(2**4);k16
37. k16 is GF(16); k16 is GF(16,'a'); k16 in Fields()
38. k16.base_ring(); k16.prime_subfield();
39. k16.gen(); k16.multiplicative_generator()

40. k16.list()
41. k16.modulus(); k16.gen().minimal_polynomial()
42. k16.degree(); k16.characteristic()
43. k16.cardinality()
44. QQ in Fields(); RR in Fields()
45. ZZ in Fields(); ZZ in Rings()
46. for a in GF(9): print(a)
47. k.<a> = FiniteField(9); k.modulus()
48. a;a^2;a^3;a^4, is a a generator?
49. a^5; a^6; a^7
50. a.multiplicative_order()