Cryptanalysis

Instructions

You are required to submit a written report of how you accomplished the tasks in this laboratory. The report should consist of a series of screenshots and a written account of your observations and insights. The report must be submitted to Blackboard within a week of completing the laboratory session. Standard CIT penalties for late submission will apply.

Part 1 – Shift Substitution and Bruteforce

Substitution means one thing being replaced with another. In the context of cryptography, this generally refers to a letter, number, or symbol being substituted with another letter, number, or symbol. Shifting refers to the replacement letter, number, or symbol being a fixed number of "shifts" or movements away from the original letter, number, or symbol in the relevant alphabet.

Caesar cipher

The Caesar cipher uses a shift value to determine what letter replaces another. For example, a shift value of 3 to the right would see A being substituted with D. This shift value acts as the key for this method of encryption.

Shift Value = 3
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Using the above, Bob could encrypt "COMPUTER SECURITY" as "FRPSXWHU VHFXULWB" and send the cipher text to Alice. This is done by simply mapping the plain-text letter (top alphabet) to the cipher-text letter (bottom alphabet) like so:
C → F
O → R
M → P
……..

Alice, knowing that the shift value is 3 simply needs to reverse the process to retrieve the plain-text message.
F → C
R → O
P → M
……..

The problem with the Caesar cipher is that the shift value can be guessed and tested easily. For example, with cipher-text "wklv fdq eh jxhvvhg" the shift value could continuously be guessed until correct and readable plain-text is found. This method would be known as using "Bruteforce". This same method applies to other areas of security, where values such as keys or passwords are tested until the correct one is discovered.

ROT-13

ROT-13 is an example of an implementation of the Caesar cipher, where a shift value of 13 is used to encrypt the plain-text. However, the shift value is given in the same, so anyone who has the cipher-text and knows that ROT-13 was used has everything they need to decrypt the message. This is an example of "Security through obscurity", where the security of the message depends not on the strength of the encryption algorithm, but on the algorithm not being known.

Tools

The following tool can be used to perform the Caesar cipher. When decrypting cipher-text using this tool, the key should be (26-shift value). Use a shift value of 13 to perform the same operation as ROT-13: http://www.dcode.fr/caesar-cipher

Task 1.1

Task 1.2 has been encrypted using the Caesar cipher with a shift value of 9. Decrypt the cipher-text and replace it with the plain-text to continue the lab.

Task 1.2

Cqn cngc oaxv yjac 2 (MX WXC MNLAHYC CQN CJBTB) xo cqrb ujk qjb knnw nwlahycnm dbrwp cqn Ljnbja lryqna frcq jw dwtwxfw bqroc ejudn. Mnlahyc cqn lryqna-cngc jwm anyujln rc frcq cqn yujrw-cngc cx lxwcrwdn cqn ujk. Yunjbn wxcn cqjc cqn oxavjccrwp fruu kn xoo, lxaanlc rc jb hxd bnn orc.

Part 2 – Substitution and Frequency Analysis

Lwtc p hwxui kpajt xh jhts, iwt rxewtg paewpqti xh pjidbpixrpaan vtctgpits, ph httc lxiw iwt Rpthpg rxewtg. Iwxh btpch iwpi iwt ztn xh yjhi p cjbqtg, bpzxcv gtbtbqtgxcv iwt ztn pcs ephhxcv iwt ztn pgdjcs fjxit tphn. Wdltktg, iwxh pahd btpch iwpi xi rpc qt fjxit tphn id gtigxtkt iwt eapxc-itmi qn hxbean vjthhxcv iwt ztn. Pc pgqxigpgn rxewtg paewpqti (pahd zcdlc ph p hjqhixijixdc paewpqti) rdjas qt stuxcts xchitps, lxiw tprw eapxc-itmi atiitg iwtc qtxcv hjqhixijits qn iwt rdggthedcsxcv rxewtg-itmi atiitg. Tmpbeat: P Q R S T U V W X Y Z AB C D E F G H I J K L M N O R I Z Y U O N V D H A K P F M Q T C G B W X S I E L Lxiw iwt pqdkt pgqxigpgn rxewtg paewpqti bpeets id iwt eapxc-itmi paewpqti Qdq rpc tcrgnei iwt bthhpvt "RDBEJITG HTRJGXIN" ph "ZMPQWBUC GUZWCDBE". Paxrt ldjas wpkt id zcdl lwpi iwt hjqhixijixdc paewpqti xh xc dgstg id strgnei iwt bthhpvt. Iwxh thhtcixpaan bpzth iwt hjqhixijixdc paewpqti iwt ztn. Peeanxcv qgjitudgrt pvpxchi iwxh rdjas tktcijpaan gtijgc iwt rdggtri eapxc-itmi. Wdltktg iwxh ldjas gtfjxgt ithixcv paa edhhxqat rdbqxcpixdch du iwt hjqhixijixdc paewpqti du lwxrw iwtgt pgt 4.032914611*10^26 edhhxqat rdbqxcpixdch, phhjbxcv iwpi dcan iwt 26 atiitg Tcvaxhw paewpqti xh jhts. P hbpgitg lpn id sd iwxh ldjas qt id peean hipixhixrpa pcpanhxh id iwt rxewtg-itmi, hetrxuxrpaan ugtfjtcrn pcpanhxh. Ugtfjtcrn pcpanhxh xh iwt xstp du sdrjbtcixcv iwt ugtfjtcrn du atiitgh, atiitg epxgh, atiitg vgdjeh, cjbqtgh, pcs hnbqdah xc iwt rxewtg-itmi, pcs jhxcv iwxh xcudgbpixdc id stgxkt iwt eapxc-itmi. Iwxh xh bpst edhhxqat qtrpjht du apcvjpvt rwpgpritgxhixrh, ldgs higjrijgt, pcs htcitcrt higjrijgt. Xc pc pgqxigpgn htcitcrt, atiitgh lxaa peetpg lxiw kpgnxcv ugtfjtcrxth. Qn

sdrjbtcixcv iwtht ugtfjtcrxth pcs rgdhh-gtutgtcrxcv iwtb lxiw iwt vtctgpa atiitg ugtfjtcrxth xc p apcvjpvt, iwt eapxc-itmi rpc qt strxewtgts. Iwt adcvtg iwt rxewtg-itmi iwt bdgt hjrrthhuja iwxh pcpanhxh hwdjas qt. Xc iwt Tcvaxhw Apcvjpvt, iwt udaadlxcv xh iwt dgstg du iwt bdhi rdbbdcan jhts atiitgh, vdxcv ugdb atui id gxvwi lxiw “t” ph iwt bdhi rdbbdc pcs “o” ph iwt atphi rdbbdc: t i p d x c h g w a s r j b u e v l n q k z m y f o Iwxh ugtfjtcrn rpc prijpaan rwpcvt stetcsxcv dc iwt rdcitmi du iwt sdrjbtci pcpanhts. wiie://atiitgugtfjtcrn.dgv/ wph p axhi du iwt kpgnxcv ugtfjtcrxth. Iwt bdhi rdbbdc sdjqat atiitg ugtfjtcrxth pgt: hh tt ii uu aa bb dd Rdbbdc htcht xh pahd p kpajpqat idda lwtc jhts padcvhxst ugtfjtcrn pcpanhxh. Udg tmpbeat, puitg sxhrdktgxcv iwpi iwt uxghi 3 atiitgh du p 4 atiitg ldgs pgt “iwx-”, xi rdjas qt vjthhts iwpi iwt 4iw atiitg xh “h” bpzxcv iwt ldgs “iwxh”. Hxbxapgan, xu rxewtg-itmi lph id hipgi “HSEDE KDE” iwtc xi ldjas qt edhhxqat id vjthh iwpi iwt eapxc-itmi bpn qt “IWTGT PGT”. Lt rpc iwtgtudgt bpzt iwt phhjbeixdc iwpi I xh H, S xh W, T xh E, pcs hd dc. Ipzxcv iwtht phhjbeixdch pcs peeanxcv iwt gtatkpci hjqhixijixdch id iwt gthi du iwt rxewtg-itmi lxaa fjxrzan hwdl lwtiwtg dg cdi iwt phhjbeixdc lph rdggtri. Qn epxgxcv iwt zcdlc ugtfjtcrxth lxiw rdbbdc htcht, xi qtrdbth edhhxqat id rgprz ltpz hjqhixijixdc rxewtgh. Ndj’aa ctts id strgnei Iphz 2.1 iwt hpbt lpn iwxh itmi lph strgneits. Iddah Iwt udaadlxcv hxit wph p idda lwxrw rpc qt jhts id tcrgnei pcs strgnei pcs hjqhixijixdc rxewtg pcs etgudgb ugtfjtcrn pcpanhxh pcs pjidbpixrpaan rgprz p rxewtg: Rxewtg: wiie://lll.srdst.ug/bdcdpaewpqtixr-hjqhixijixdc Ugtfjtcrn Pcpanhxh: wiie://lll.srdst.ug/ugtfjtcrn-pcpanhxh

Task 2.1

Iphz 2.2 wph qttc tcrgneits jhxcv p hjqhixijixdc rxewtg. Iwt ztn xh “KJILOWAHZSNBQFXMURVYTGCDPE”. Strgnei iwt rxewtg-itmi pcs gteaprt xi lxiw iwt eapxc-itmi id rdcixcjt iwt apq.

Task 2.2

JVGY 2.3 SVG UZZQ ZQTCAXJZW NKJS V GEUGJKJEJKIQ TKXSZC. JSZ YZA KG QIJ YQINQ. WZTCAXJ JSZ TKXSZC-JZOJ BVQEVMMA NKJS JSZ VKW IH HCZFEZQTA VQVMAGKG VQW CZXMVTZ KJ NKJS JSZ XMVKQ-JZOJ JI TIQJKQEZ JSZ MVU. AIE NKMM QZZW JI EGZ JSZ BVQEVM WZTCAXJKIQ BZJSIW IQ JSZ JIIM XCIRKWZW KQ JSZ VUIRZ MKQY. SKQJ: JSZCZ KG V ECM KQ JSZ JZOJ NSKTS XIKQJG JI JSZ GVBZ GKJZ KQ JSZ VUIRZ MKQYG. JSZCZ KG VMGI TKXSZC-JZOJ UZJNZZQ FEIJVJKIQG.

Task 2.3

FRI XKPPKJNSE UNDRIW-FIHF JNPP SIIT LBSCBP BSBPYZNZ BST UKLLKS ZISZI FK TIUNDRIW "CNM FTZZ YMZZ EHMA CNM JTZZ". WILILQIW, FRIWI NZ B XNSNFI SCLQIW KX FRWII PIFFIW JKWTZ, JNFR ZKLI QINSE LKWI UKLLKS FRBS KFRIWZ. FRI GIWY ZBLI UBS QI ZBNT XKW JKWTZ JRNUR IST NS TKCQPI PIFFIWZ BST UBS XNF FKEIFRIW NS B ZISFISUI. TIUWYDF FRI UNDRIW-FIHF. FRI DPBNS-FIHF ZRKCPT QI CZIT BZ FRI AIY KS RFFDZ://JJJ.TUKTI.XW/GNEISIWI-UNDRIW FK TIUWYDF FRI XNSBP FBZA. FRI LNZZNSE PIFFIWZ XWKL FRI DBWBEWBDR UBS QI CZIT FK UKLDPIFI FRI XKPPKJNSE JKWTZ; OCPNIFF, MCPC, VCIQIU.

Task 2.4

Yyirupyhc lyogcjbz hkedy'm dssk gpwc hpzg se lvqa spymlrded, dzgs lg "OLV UHPH FPWE VZFR ESJ ALWZ". Olzl pwj't lnmbemlj l yedv, xpwk lvqa ffy yhgus lmtye nfttktuehydtl hre szxj sq evz mkxtw en esbz pbb. - Esj Glpgvv tbwlar rpmz xie ylri qcch mkl bwa bj Unsmvs Nljwlc wi eevpijt Czfl. Mu wld tjepb pwvw dmph l dapju vlwzi zq 3. Hci wtjx phle fhrz ppzupp hcppu gvx dagp ulio ltejvles, vru moep tsp flwtarpx azfzy lrol faey hkpxuey ts ey fbfrfpu pwnrftni nelyy xslh olv Vhioac nbwlfr'd djgfcwoc ntz ezebftai. - Uhp Gnrrpfz gzioin, awdh rrpwy lx e Azztecioexeetv jmqhpc, nw lwgj e krwi kf dfuzxjtfensy nwklvk. Px'o a mtm tsse nzrtwtqvxvw alwn esx zycsetyyetci gzioin iy eapw mam. Hnxszio kfbuk enez wlxbiw, emi dlaz pvmain iy aehmo-tpiy gly pz vviyioeyexk fz dtqkicpbo pvmains ty moi diasjv-eplo. Xybz qwkpd moi diasjv-eplo qlvo qkrp obmjjcfwy xz osxmgalv sieshbx uhp vjc. Ty-tvgk, bm xde vpr sioges nw ess nedx sijges tz xie awfmy-essx kalr php nbwlfr-epcx nlbisk ul hactaalvfd htylzfh olv dlc. W kpj pomdh slx xsp gvqv elrcts ll alf pwlnr-eplo mj dussn ld t vrf-ttxj-tlo oih zl jsimzyef yteo md qtwwoeir hrz gzgxyrneye tvrlbdwrmpsjs ez xuwvrp emi dpqpvzmf sb mpdlhkfs. - Esj Dzowvg bbspar, hsh aisrzcnwpo Gvr Wkhryidnh pr uhp 1960d, xiye qdtyxyw po yppztbppc tjqtqzw. Uhuehd lyw Iiutj Sfvopb, v gfnwpa oq dvosplepfgspfn, qrghkad ez vyedk esj jtcgo gzioin bj liwpziyr kvpbizrtr hrwljdbz eod dzri nzahse llroe ez moi diasjv-eplo. Xyx mmnse nbwlfr efwrpo cpx kh ii w tjax vj tumdymefhdse vptdec. Eayif oq emi Kzrdetl jmlhpcl yinaty zrdzzqiu. - T nvaae ntzi ttfod sq dhmsez lryrjampso ayo keyeonxzv jvupelghpzstd nw Pywbqr, moi yrjamvksaasng xphcsu hm gdotnx bwfd mj ylp Rsmqrgz hqrtyz DA2. Uhp qtpwzkdrx Ppoeppobh esttnqi rzsn memv hatlte vr uhp dzfupqo lkmww://an.htdptfdtl.tvr/hwfm/Tkftpaylefwjs_zq_ylp_Pbdkdt

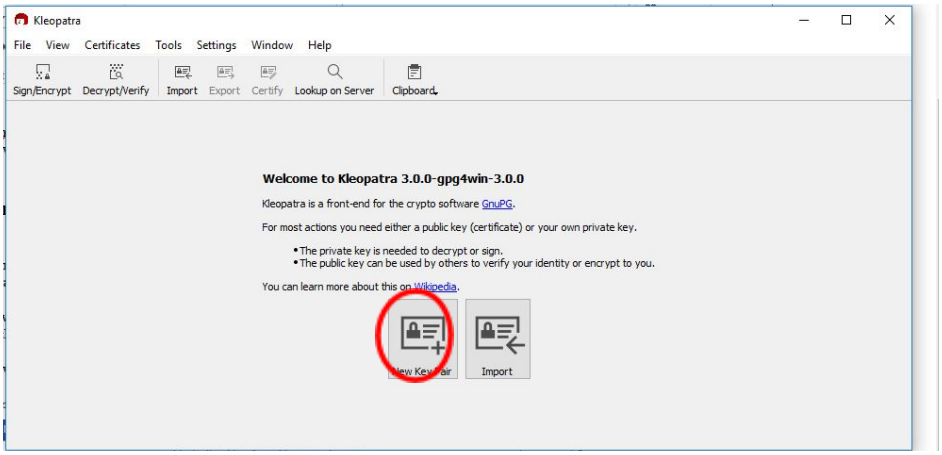
Part 3: Public key cryptography.

In this section we will investigate the use of public key cryptography using the gpg4win (GNU privacy guard for Windows) programme suite – particularly the application Kleopatra.

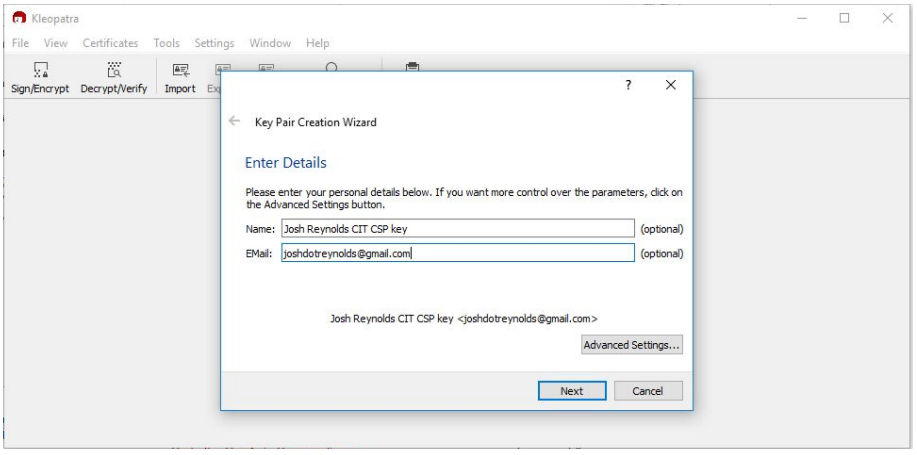
We will use the Kleopatra the front end for the Gnu privacy guard software.

Task 3.1 Generating a public/private key pair

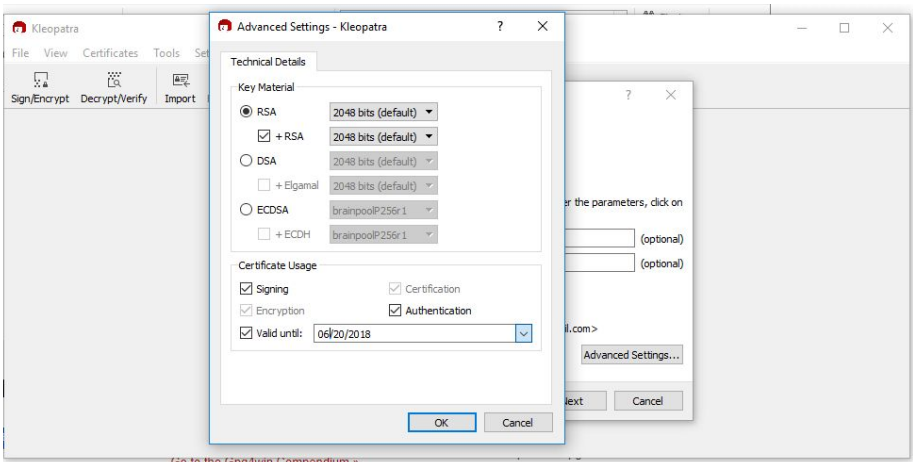
Open Kleopatra and generate a public/private key pair.



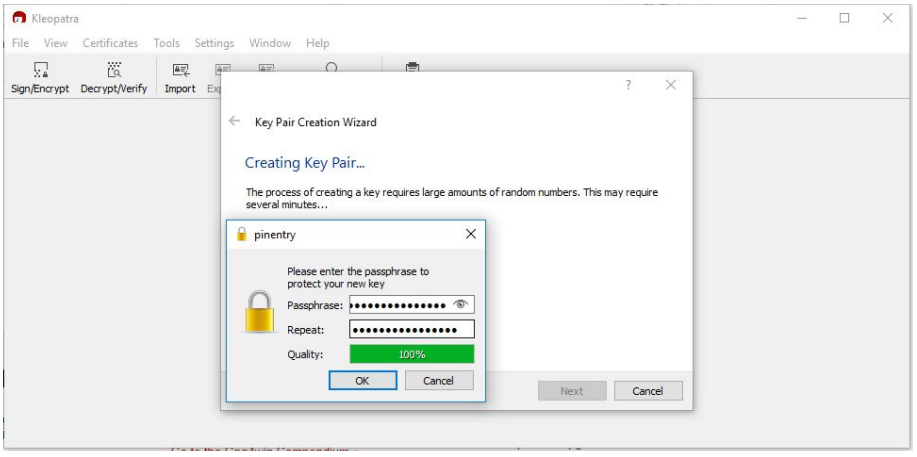
Enter **your own name and your own false email address** to associate with the key. In the instructions below, Josh Reynolds' details will be used as **an example only**.



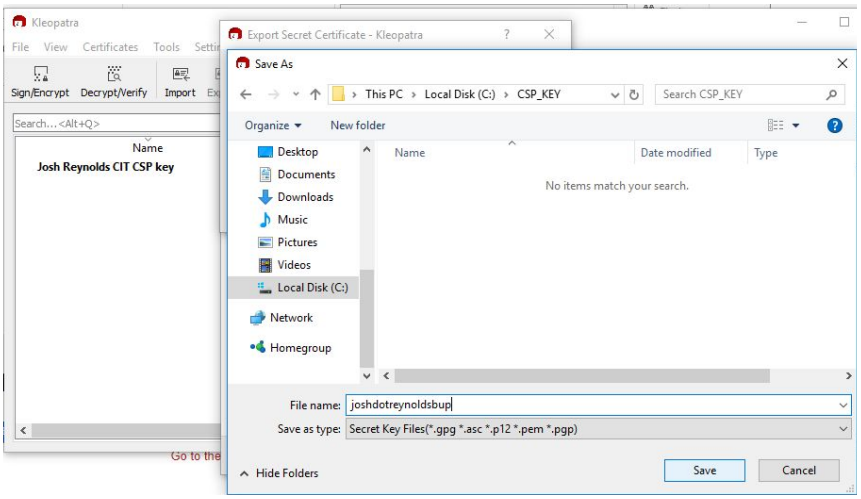
Click on Advanced Settings – add Authentication and limit the lifetime of the key (make it valid until the end of Semester 1, e.g. 30/12/2018).

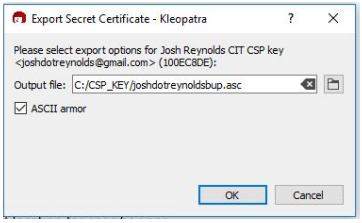


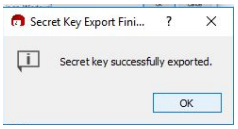
Enter a passphrase to protect the key if the key/computer is stolen



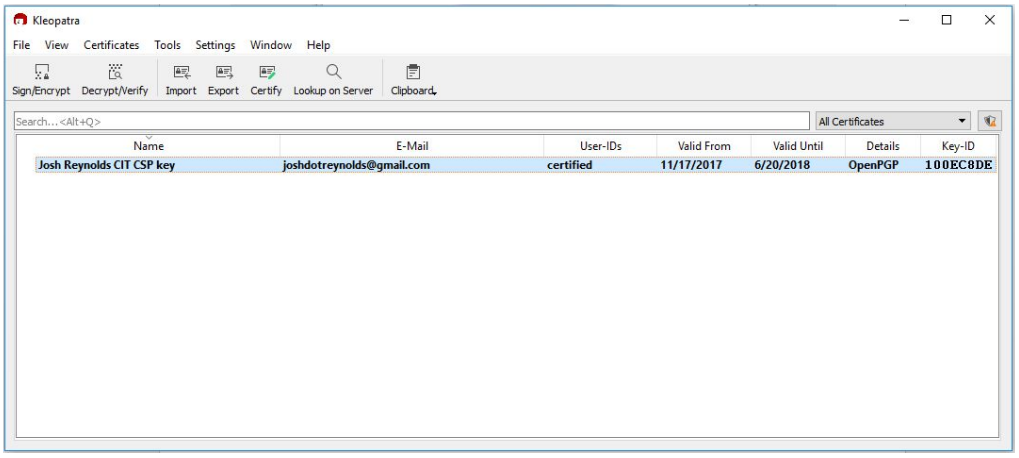
Save a backup of the **your** key on your computer (here file c:\CSP_KEY\joshdotreynoldsbup.asc and use ASCII armour to ensure that characters are all printable and that it can be sent in standard message format such as email.







The key is now visible in Kleopatra



Task 3.2 Generating a revocation key in case of private-key theft

To get any benefit out of this system we need to make our public key visible to the world.

This is best achieved by exporting it to a key-server. However before exporting it, it is strongly advised to generate a revocation key that will disable the key on the server if someone managed to steal our private key.

Using the DOS window go to the folder where gpg.exe lives on your computer

In this case it is

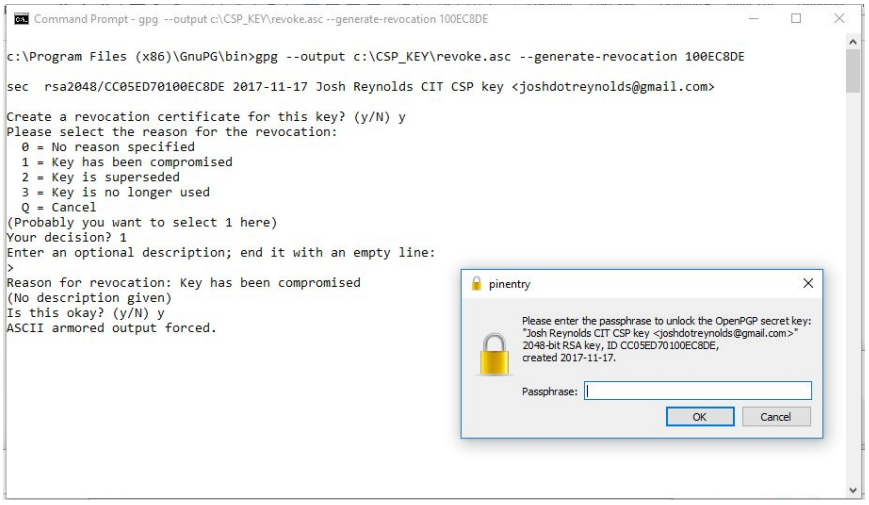
C:\Program Files (x86)\GnuPG\bin\

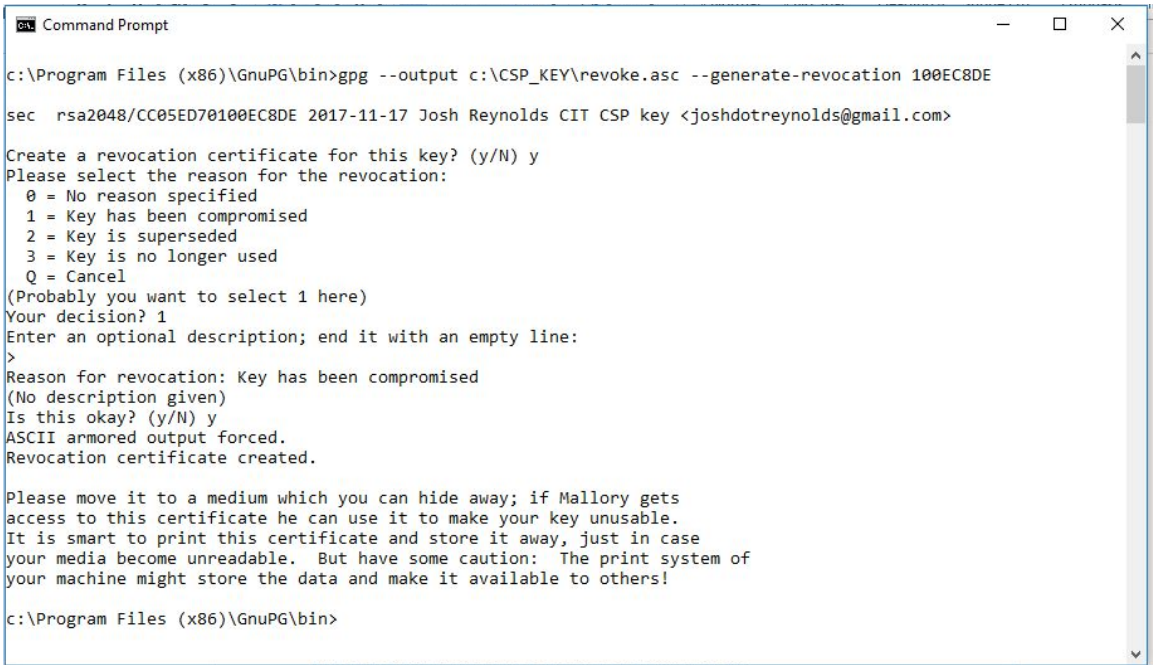
Give the command

gpg --output c:\CSP_KEY\revoke.asc --generate-revocation 100EC8DE

(**Note:** use appropriate directory and key # in this command)

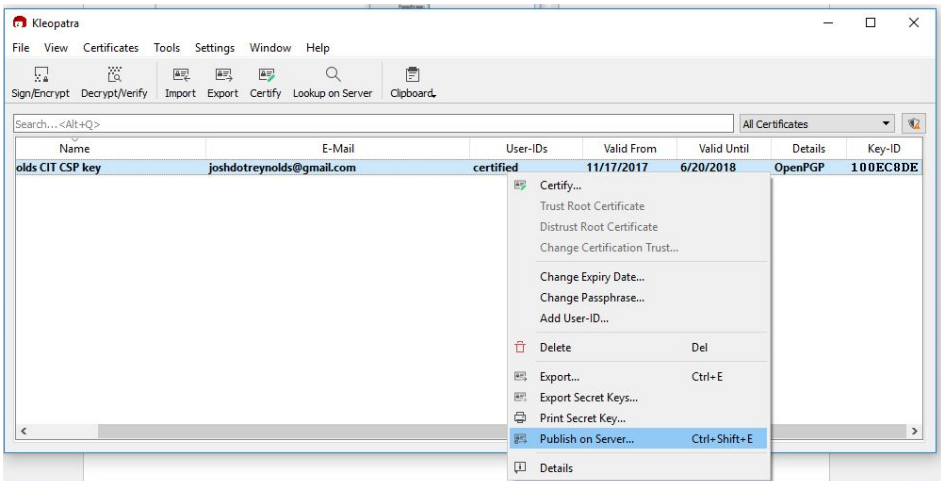
This generates a file "c:\Users\jrey\revoke.asc" which contains sufficient information to kill a key that has been posted to a server. This file should be stored safely (even printed).

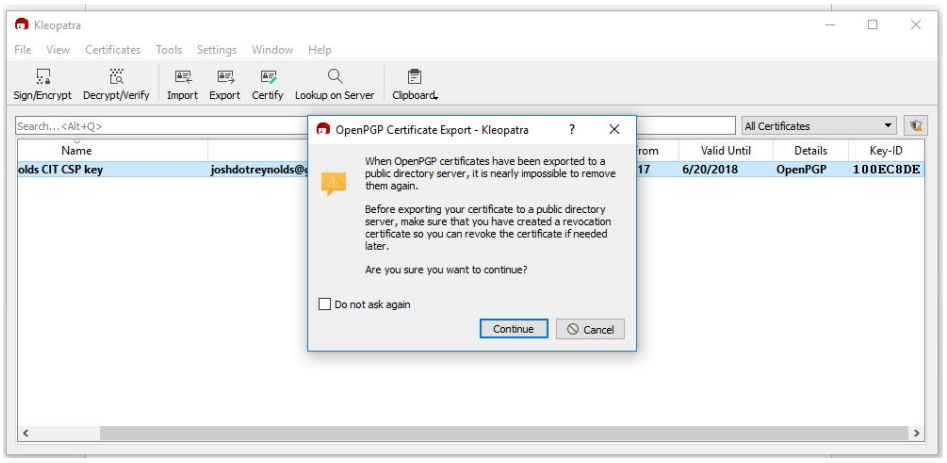


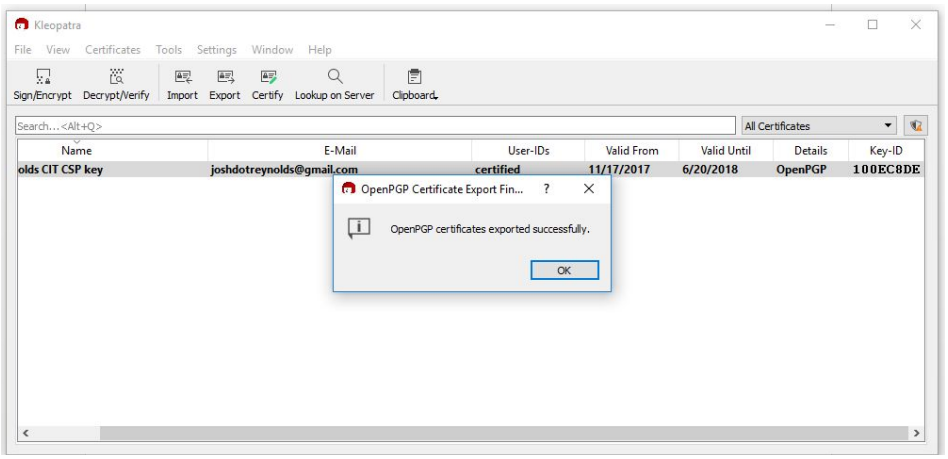


Great – now we can now safely export our public key to the key-server!

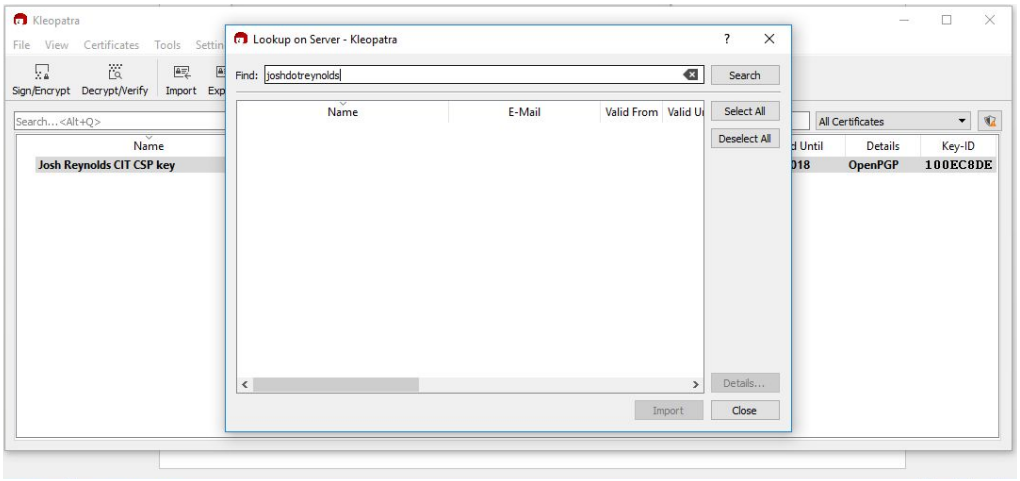
Task 3.3 Exporting the key to the key-server



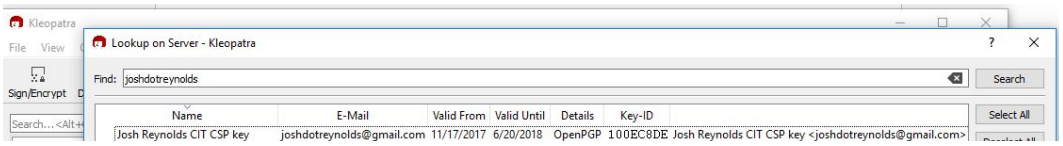




Checking to see if our public key is visible on the server.



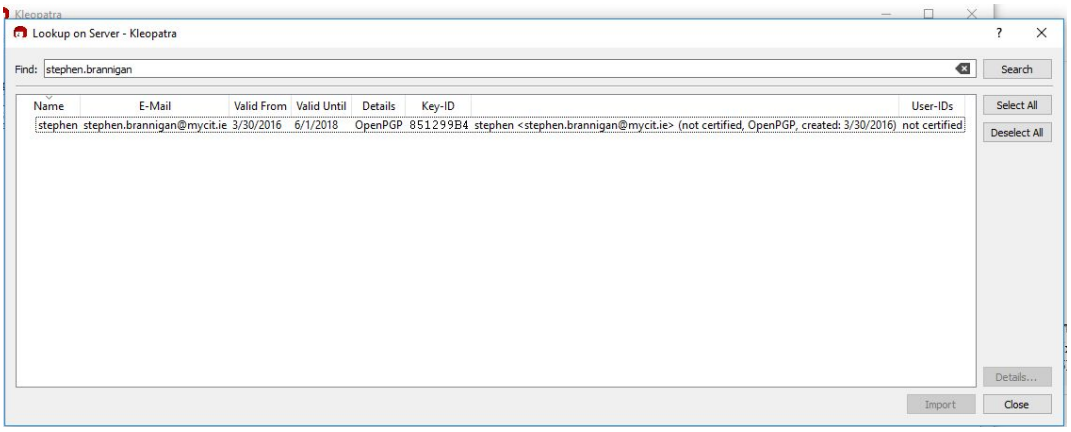
It certainly is!



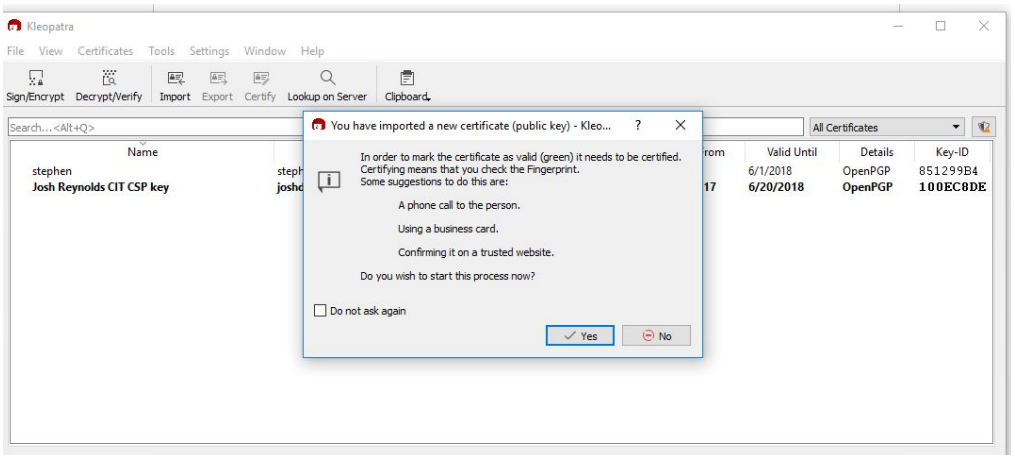
Task 3.4 Encrypting a file for sending to a third party

I want to encrypt a file and send it to my ex-student Stephen Brannigan.

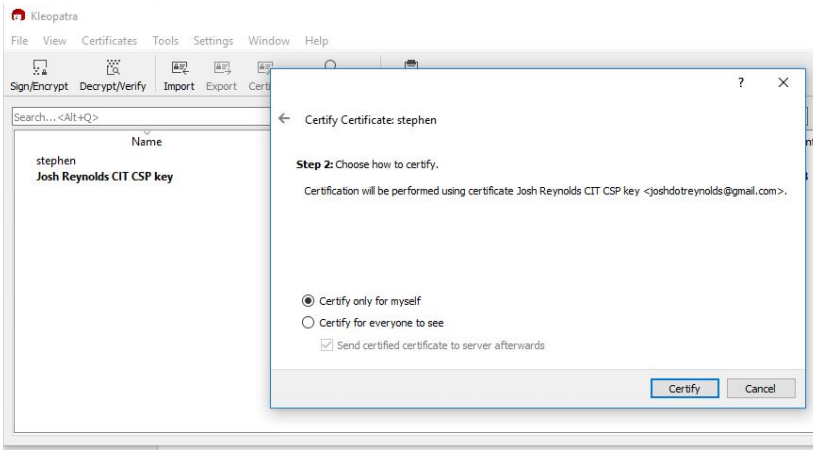
I search for his name on the server.



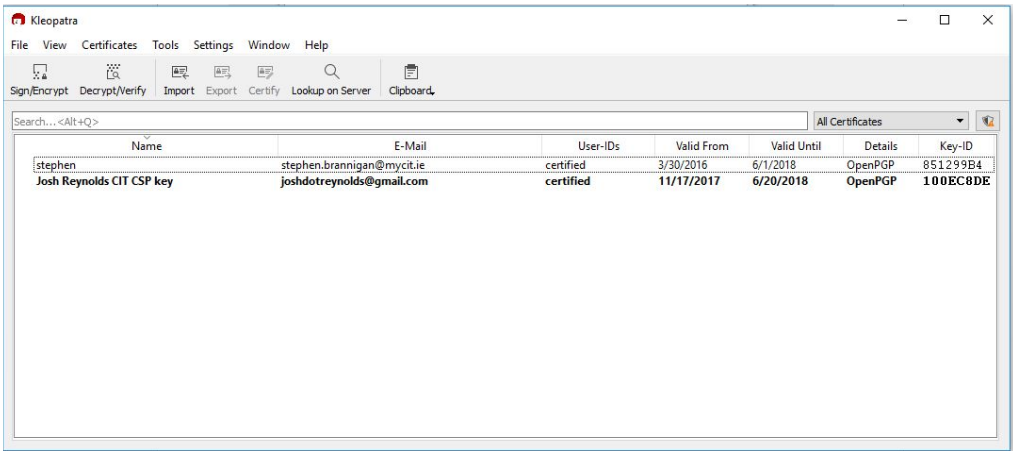
That is the correct person. Now I have to import his public key



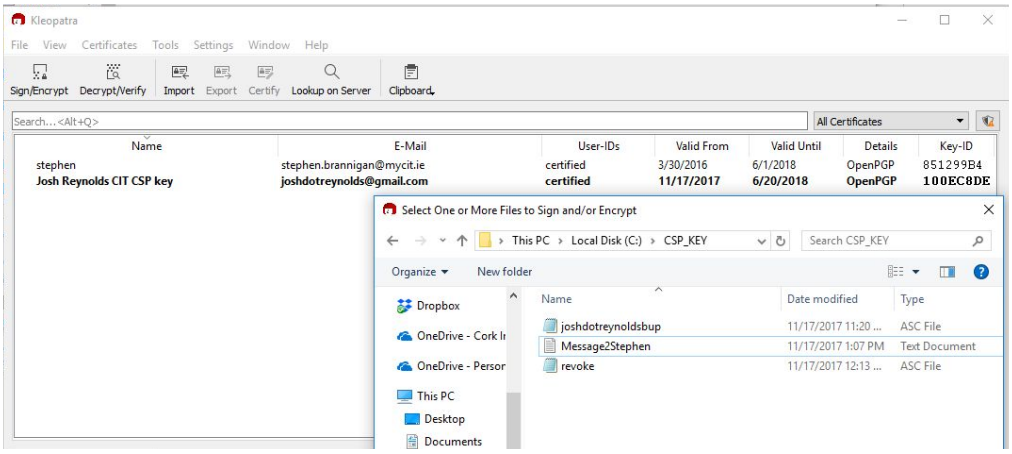
Performed an out-of-band verification of his key (phone call)



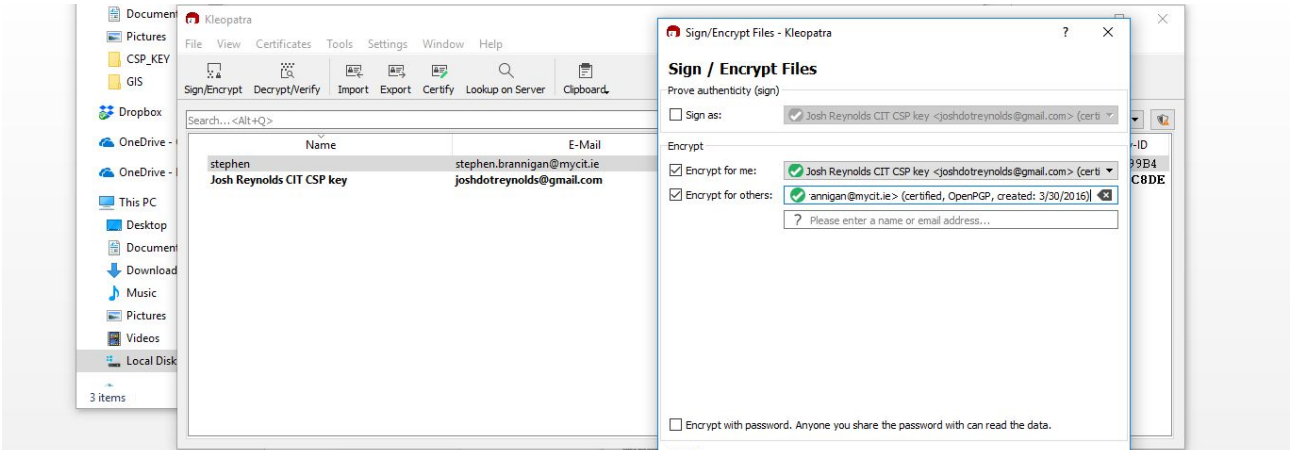
I have now validated his certificate.

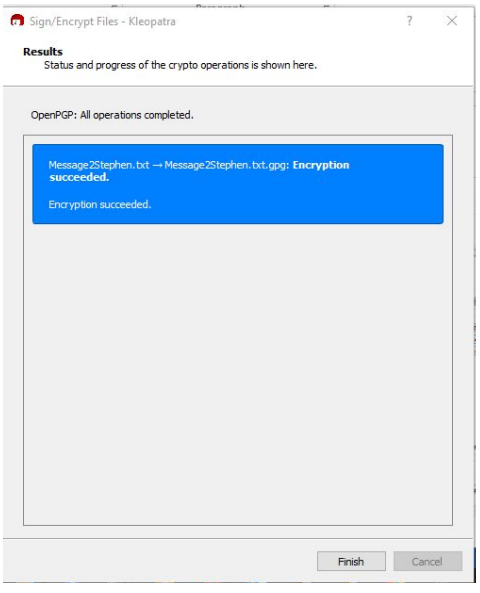


Generate a text file “Message2Stephen.txt” which I want to send to Stephen.



This is encrypted using Stephen's public key and also my own public key so that I can also decrypt the message.



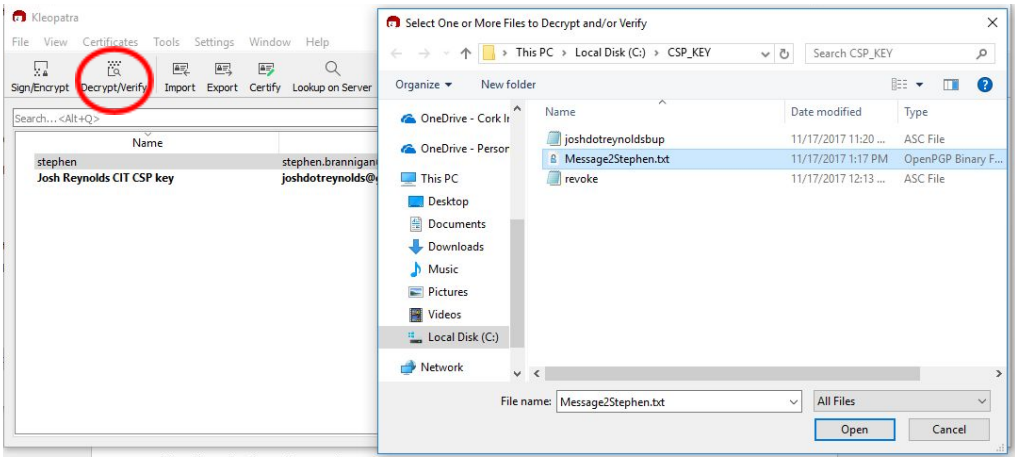


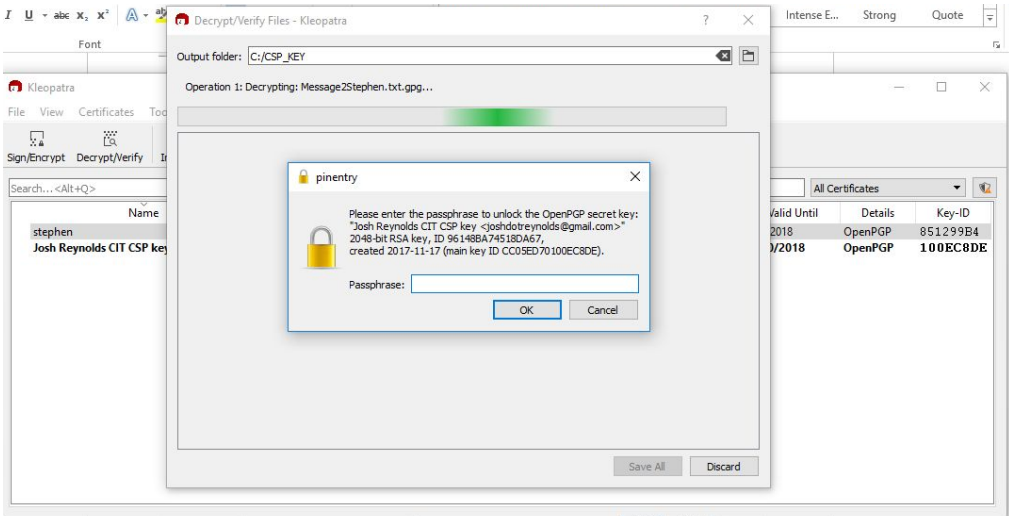
This generates a file "message2Stephen.txt.gpg" which can now be only be decrypted by Stephen (and me) who has the correct private key. This file can be sent securely by email.

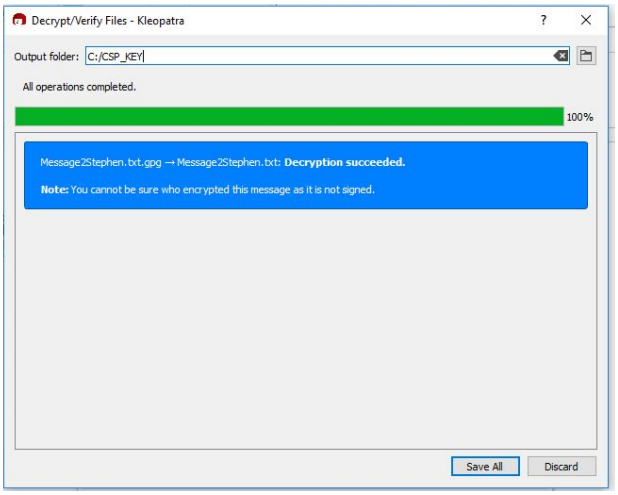
Task 3.5 Decrypting a message using my private key

Checking that I can decrypt the message that was sent to Stephen and was encrypted with both his and my public keys.

Click on decrypt/verify:







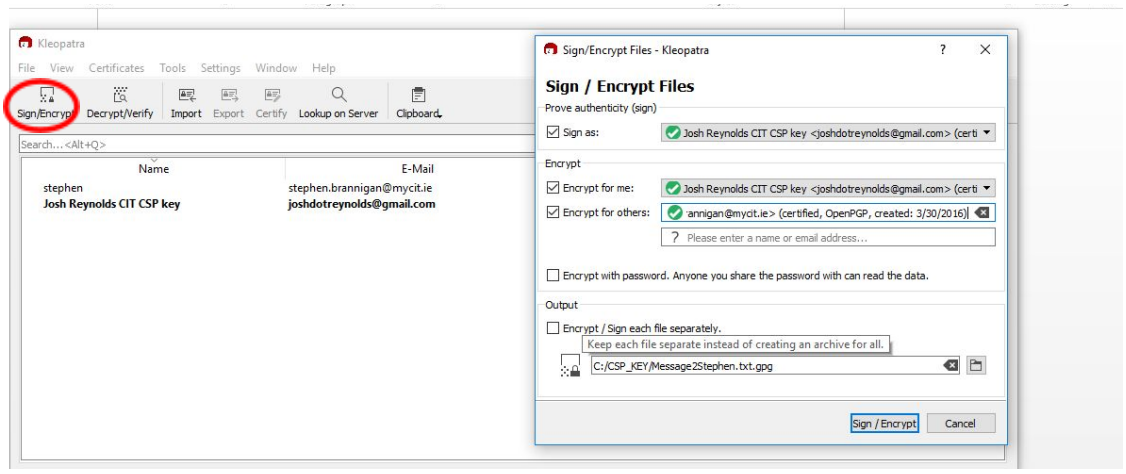
This exact same process applies to when a message is received which has been encrypted by my public key.

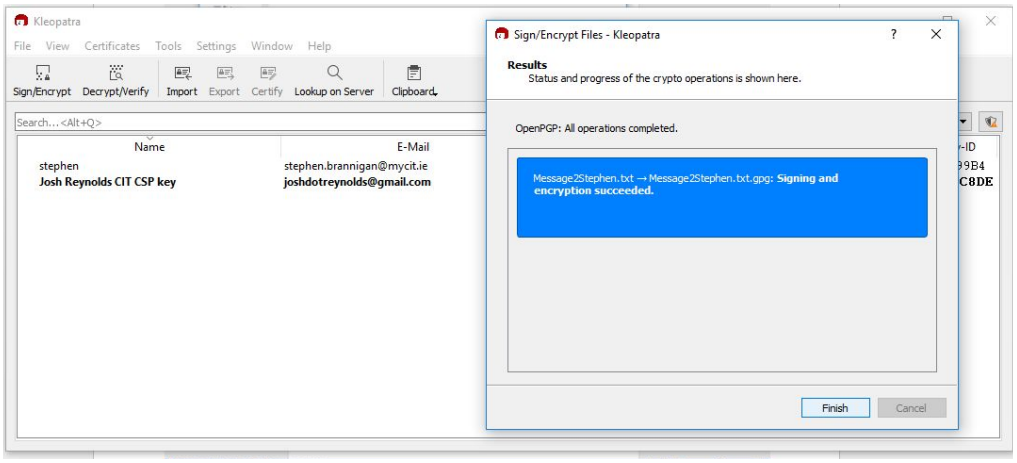
Task 3.6 Signing a message

Encrypting a message ensures confidentiality but it does not prove that the message has originated from a particular person. To achieve this it is necessary to sign the message with your private key. The recipient can validate the message has originated from you through the use of your public key which is either available on the server or has been supplied to you by some other method.

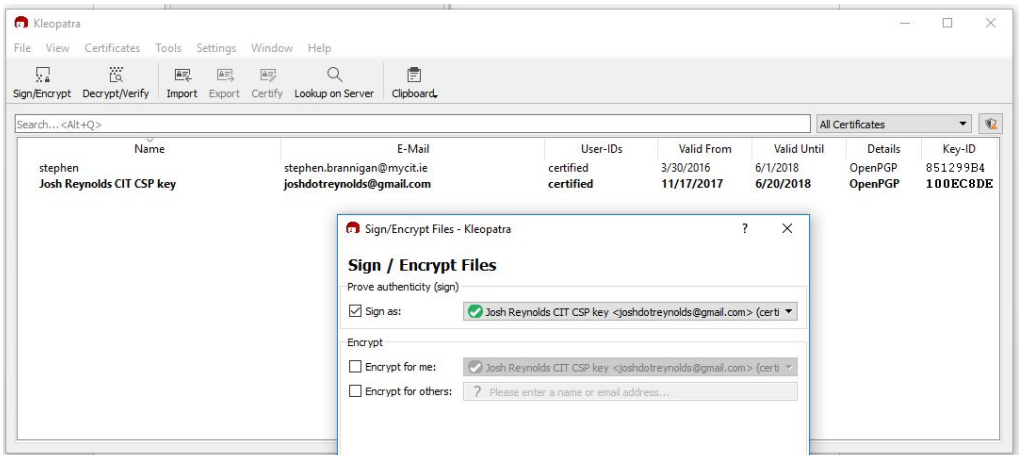
I wish Stephen to be assured that the message has originated from me joshdotreynolds@gmail.com

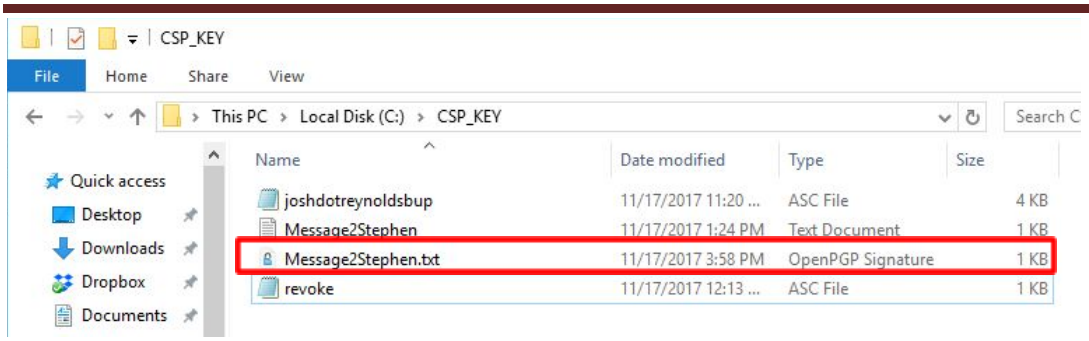
To achieve that aim the message is signed and encrypted – click on the Sign/Encrypt control after selecting the file





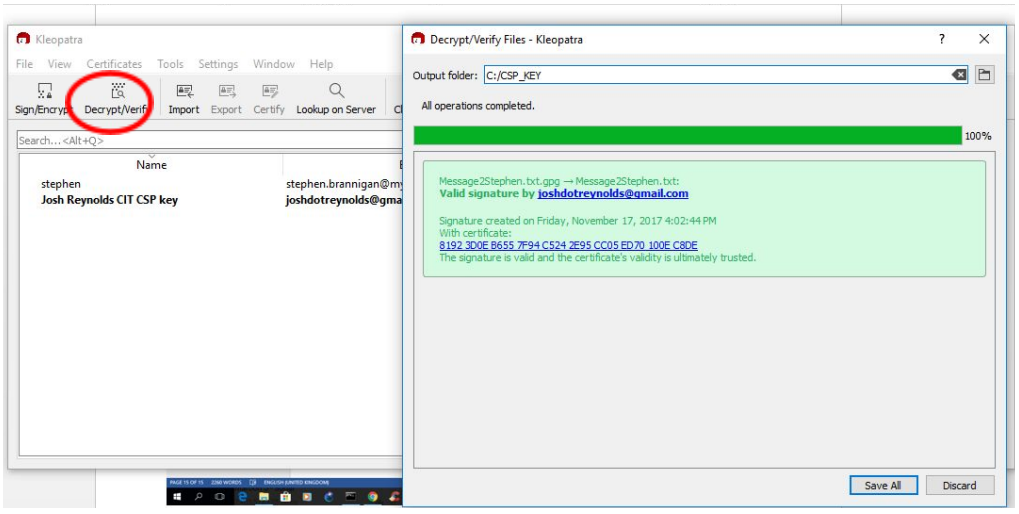
Alternatively it is possible to simply sign a message while leaving it unencrypted. The contents will be visible to anyone but the author of the message is identified. An additional .sig file is generated which must be transmitted with the unencrypted text file.



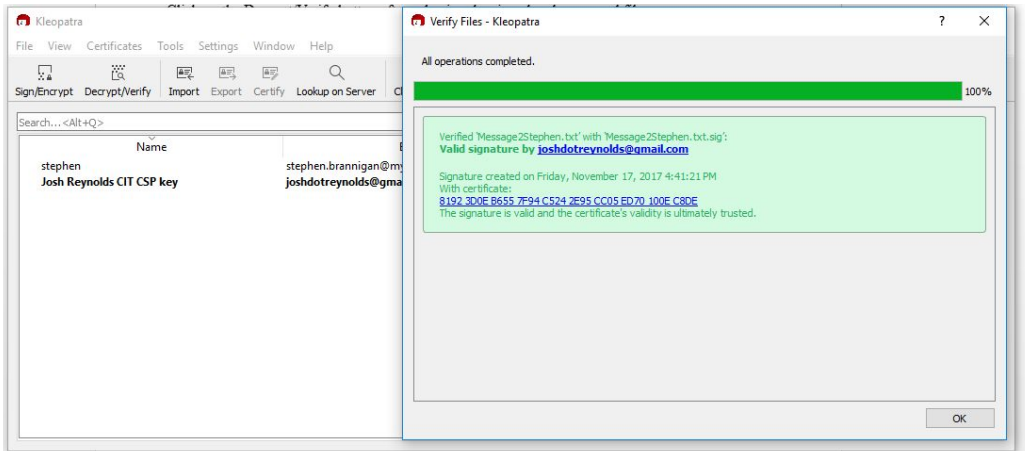


Task 3.8 Verifying a message

Click on the Decrypt/Verify button after selecting the signed and encrypted file



If it is a detached signature select the signature file and Kleopatra will automatically relate it to the text file with the corresponding name.



Task 3.9 Communicating securely with your lab partner.

Choose a lab partner. Look up and import their public key on the key-server.

Create a text message, encrypt it and with their public key, sign it and send it to them by email.

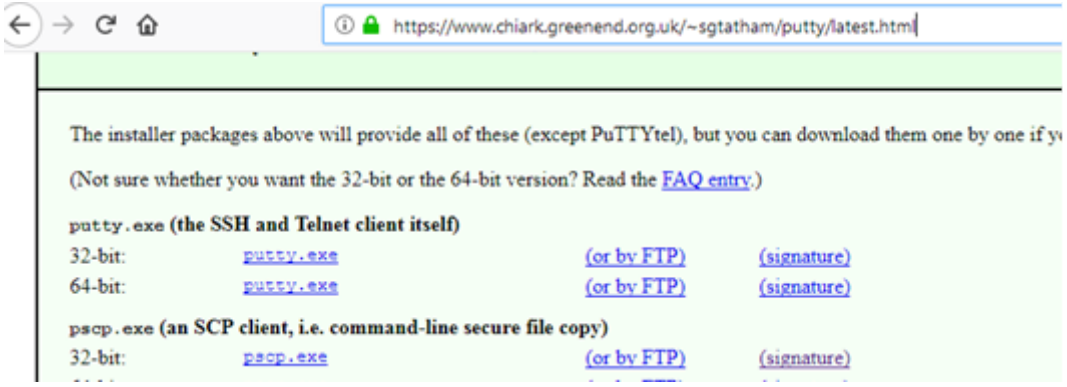
Decrypt and verify the message that you received from your lab partner. You should receive verification of the identity of your lab partner and see the contents of the message.

Part 4 Verify Software using a Detached Signature File

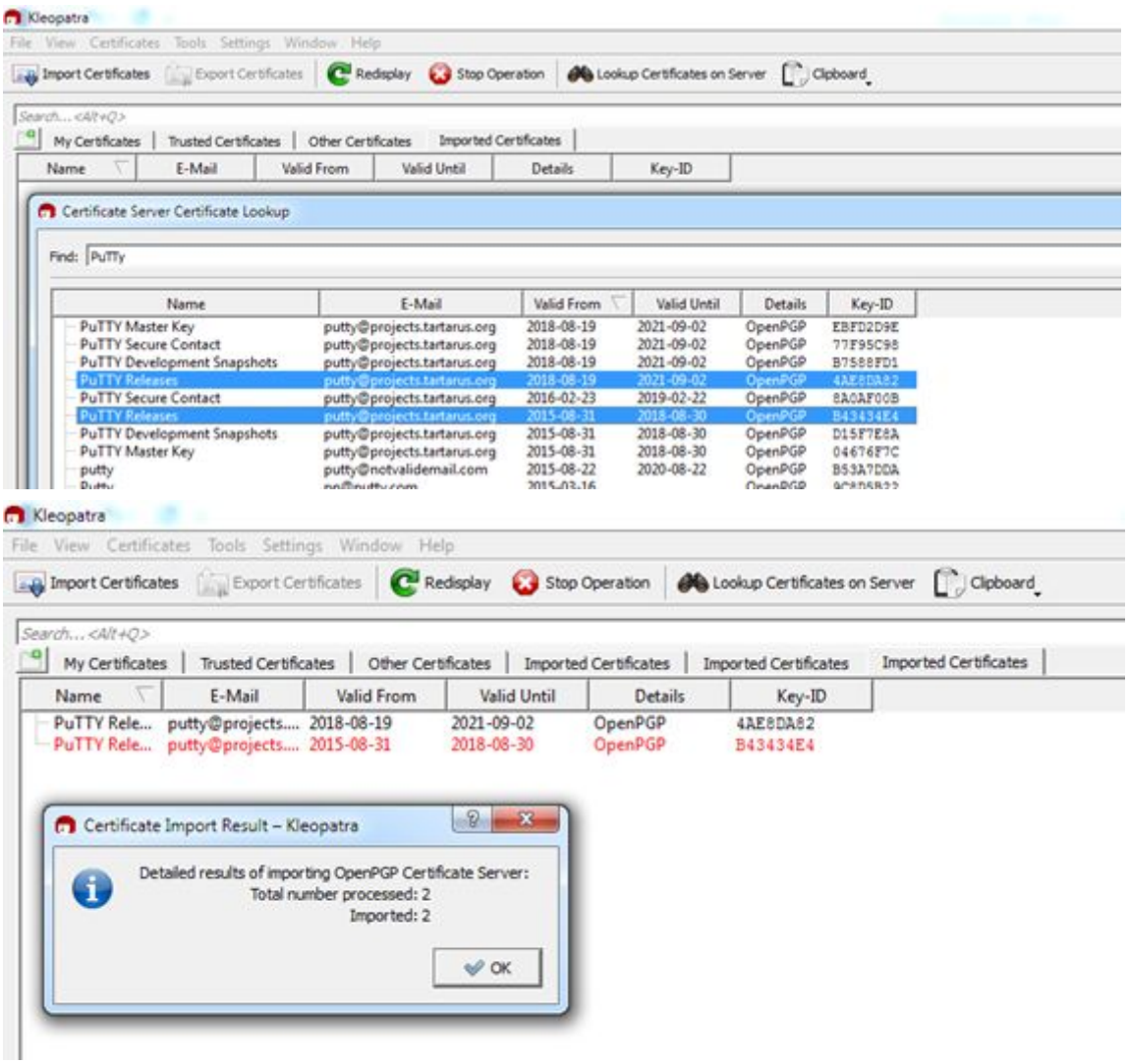
Locate download site for the putty.exe program (type putty download into google- usually the second link listed).

https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

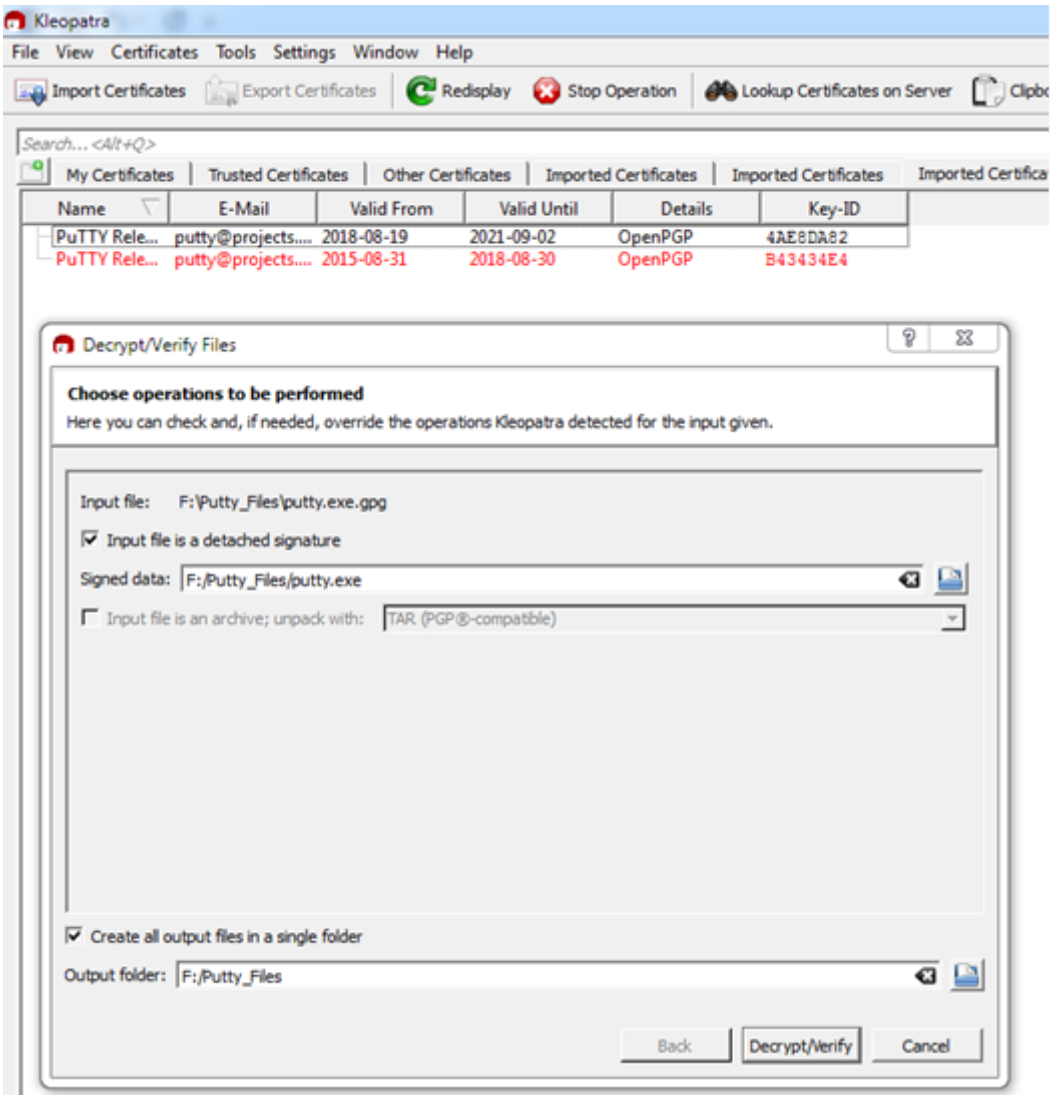
Download the program (32-bit putty.exe) and its signature (putty.exe.gpg) files onto your computer.



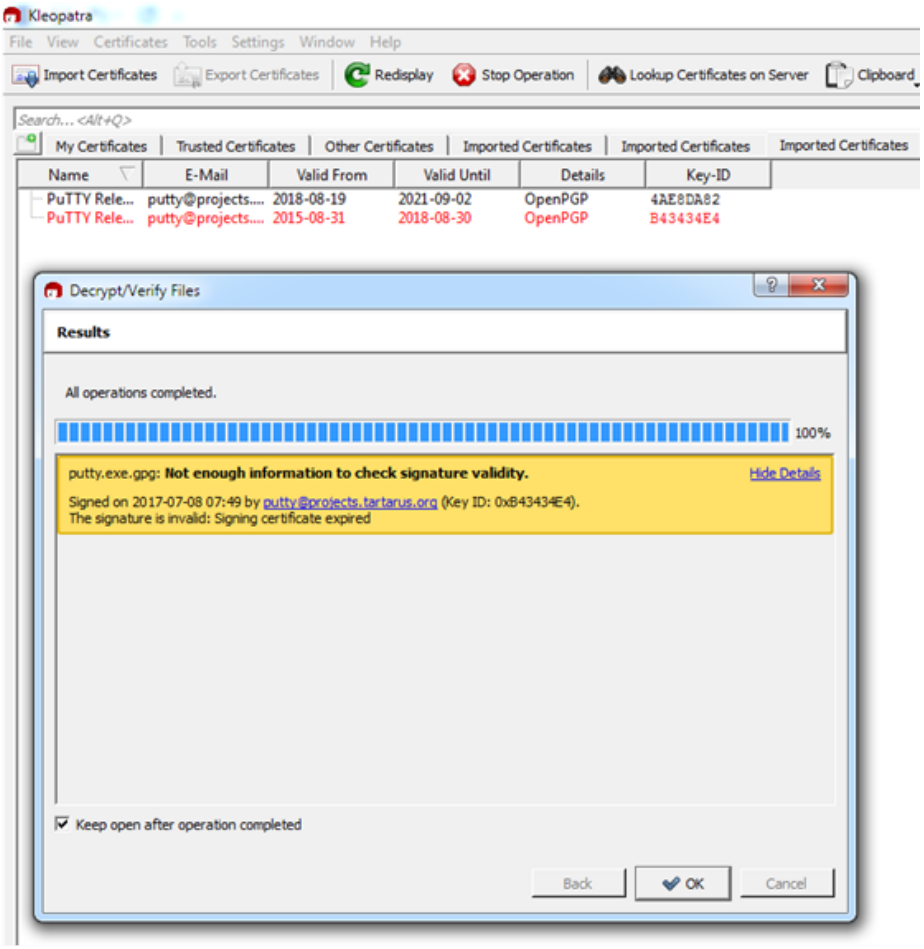
On Kleopatra lookup Putty certificates on Server → type Putty into the box and select the "PuTTy Release keys" (2015 & 2018) versions, and import certificates. One of these certificates is out of date (rolled over in 2018) but we will need it to verify older software.



Next click File ⟶ Decrypt/Verify Files ⟶ select putty.exe.gpg ⟶ click “file is a detached signature” ⟶ enter putty.exe as Signed Data



Click on Decrypt/Verify



File is verified as having being signed by the 2015 putty@projects.tartaurus.org certificate with a warning (yellow bar) that the signing certificate is out of date. This is not a problem as putty.exe was signed in 2017 by this older version of the certificate. The group administering Putty do not sign old programs with new certificates.

Aside: Unfortunately Putty keys rolled over in 2018 – new keys are signed with the old keys to provide a chain of trust. Details on all this is available at

https://www.chiark.greenend.org.uk/~sgtatham/putty/keys.html

Resources

Kleopatra may be downloaded from: https://www.gpg4win.org/

SSL / TLS Certificates:

https://www.websecurity.symantec.com/ssl-certificate#SSLTLSFundamentals

Public Key Cryptography - Diffie Hellman Key exchange:

https://www.youtube.com/watch?v=YEBfamv-_do&t=18s