McKenna Wirth and Antonia Ritter

CS338 Spring 2022

SSH KEY FILE FORMATS

Contents of id_rsa_homework:

```
-----BEGIN RSA PRIVATE KEY-----
MIIG5AIBAAKCAYEA1kx9ABgXb3VGFXIcYRzPC4FmXl8+4gCaQ36yrEgp9kAxq8U9
ZY+CD6kczqcqo7O0vtze5V5pg+Mbnc/TXT9L0GrJs8+9hZYXvGCGim5HWGa5OxZA
EcFDfr3YGbqUcjTmWQmxBmoLyMdmJf7sFvSfpRT5Lq88VFvXOcLwArpkFR3GNjgw
UPfG9dBF+mwgMqIOM018GNCQuafimpfVKi/mbqmCKtFOtooouREAnrW5MB8G5+OC
J8BA0V02CU4/dQuilTRC+LlwcMxantGs1TE832WzOcd7HLVQA4AcbyscLIA0eymX
5SIELINT+c5lM68sBY0MKPebsCo28ar5u9SYQNQlKIuq83scxc20GmTF67yH9GDH
3G7b4O7FCm15bp7rLgbFAKqY4K+BY0DHNP5kFz7V3Hjw9s/aI6489HsIcvmjpB1J
UxghFZawyyWhC69nXNqVuXG+eHIJ1923jB7A8DljIMLSdPVZGMV4vli83yjWtaAp
wgaUJW/tQDSGDhtDAgMBAAECggGBAMsH3qoXKSTtPAAXg7GyRKsG5wDkRkSDA6SM
xiUZnb/zo9u03G17YGML+wIDOQ+ts8HcqsUvBNLyeXizrPI8hMyqT/0vBcUppcVQ
wkcnIisr77R+aScpLpRGsxX1Qq1NVyvmVbTZBd8so6e99Nnu283gqBDF72eRN8E+
4nwDMc0bz24RIB2mEvdTYulWRc8S4FdUww0TlPqAh92lmk0E4tHXgT6l3WhuTxS/
ehHj49cmnjlWwDp/GCoDoK2CV/bBlIRumYePFYqJEv8mCYHSiEN0rlSVTQL1vahE
5QVgwrCs3OA+iEZdltpFywIGRN/iVhCSg8sNL4BhhDHkiJlctxdrNkGCq3GZMlO5
wkIVO3YoNQczLeniUqqmq8nEKRQWFNglLTV1OqZ4oRR+nYjsQ0ZB+h8YUm6J3SLx
c26jSFWIdS2Z0BjU62ibR+SyPUKL4S8u7Gh3lorcTxu/1vWUK1b5/LL44NSFTI14
tH4DBd45ngU4UdbaCw4imD6Cj7a3MQKBwQD1d8yk4MRJLtOciWyKUjRN0zadrzgO
XQ36JBuEJy+GXwRiQSM3zQg1WBLoSfFefG4nQpyOso0xKcA4unA4QYZNfetkqqW8
4eND4P6tDZikzKWMk9mTSBBYUeovhTXRd+SzXzgkptt6CRgFzp86490RH+SQkMR0
mScI8Vx77nY6M4SgR/pviGFWlvMBT7O3kAA8jeJ99ScqJp9OgRc4F97jgIxItSGN
hUqsMQuVlhK2Y2p+lOKp7Zbk5AzRTq0A6IsCgcEA335UPPHfKWXm7M40PuXeeAwM
stayIuUMBXcLhnxPoKbsOGCG9oHXZB/S+SUzHR5duZPNicxslqOHhqgm7xWD5qME
exvSj7S0YYN3LuElY2MRPaiehsOvQAa4B9Xq4MkAQr8tdZPM683/iB66PuikhpvA
eulEtgvrwUdwwAfITsutDjgUXEx/ggRrO+wPNUDai200CWvdM2z9XjgbKqXrSNmo
1Oe/gHtzs7VZOtqi3Vwf+TkBZPmmn5ponFaIOTcpAoHAf6hNClZGXpT2wksJ4qm7
v9mfZsQwT+//C97pXIIQij4yBnM/wI5lvXBPbPS/jVcC5nIS/3dGTJIdAKOdJvSU
xwo8eD7zMNb9HVkOuar/fn1bf0+I8LqinXwEYbBSe9xxWQOlLBKGw+Gk16hMUREd
8AD5GmHfeZAIp/L2GRG79KrER6MHUCfKD4EzYbQ0j7tZFCTp7H2pX5d0dBxYmfh4
FzLMQifJClOOsRxUnPfORYJQaQI/GwaP2fvr2pKjEHzfAoHAfsKVHrgE3eMZNgjC
o0YXhzk2XpzdDmOhDNx3YgDR3QmwA03kXp5IP1a5f2w08HgPqrkFI/oZQNthe+OV
qZcnZ8HgQb8HD1Dr2j8Jc0JpxVQkLDeeKkuE86ib6n3PKU1342OxcfWxjbEaFOdw
tUuYPWzLoCr8ONDMF6Tgc1o2p1JoRh9SlQMicuj0tVjvS0IMcnYuxIU0Qf24Ox9R
cUAIEhq54n/LIyFh/3SFNpdZNnoQ/HbzroTWl8GpQbrmqqT5AoHBALW44W6cSyxi
```

```
ZKF+boJ/m1twoZzlBLwY5/4R5fa8C/zKfu6DT+0KFRXpcTHUNR93PGQ1INgy7cAh
ZcmulcZn1u/8UlQVyV2BXr3aEQI4aWJucOpIu20JZK+qkxpZMs8UEwu1hoWA3EWV
r/G6u2vCkqeNS8hCts1v8t8A6hOsqJlzyP4m3Knqr3pprbKCuySmLNBcs0XThU+L
WWygPNMW6Gxp/9O0PkdFF9iYnb5FFsP74F/Um1zYv8t7TAxCa7Q4EA==
-----END RSA PRIVATE KEY-----
```

Contents of id_rsa_homework.pub:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDWTH0AGBdvdUYVchxhHM8LgWZeXz7iAJpD
frKsSCn2QDGrxT1lj4IPqRzOpyqjs7S+3N7lXmmD4xudz9NdP0vQasmzz72Flhe8
YIaKbkdYZrk7FkARwUN+vdgZupRyNOZZCbEGagvIx2Yl/uwW9J+lFPkurzxUW9c5
wvACumQVHcY2ODBQ98b10EX6bCAyog4zTXwY0JC5p+Kal9UqL+ZuqYIq0U62iii5
EQCetbkwHwbn44InwEDRXTYJTj91C6KVNEL4uXBwzFqe0azVMTzfZbM5x3sctVAD
gBxvKxwsgDR7KZflIgQsg1P5zmUzrywFjQwo95uwKjbxqvm71JhA1CUoi6rzexzF
zbQaZMXrvIf0YMfcbtvg7sUKbXlunusuBsUAqpjgr4FjQMc0/mQXPtXcePD2z9oj
rjz0ewhy+aOkHUlTGCEVlrDLJaELr2dc2pW5cb54cgnX3beMHsDwOWMgwtJ09VkY
xXi+WLzfKNa1oCnCBpQlb+1ANIYOG0M= kali@kali
```

Private Key

Items in private key file:

1. version - a version number (integer)
2. modulus - the n in the RSA algorithm (integer)
3. publicExponent - the e in RSA (integer)
4. privateExponent - the d in RSA (integer)
5. prime1 - the p in RSA (integer)
6. prime2 - the q in RSA (integer)
7. exponent1 - d mod (p-1) (integer)
8. exponent2 - d mod (q-1) (integer)
9. coefficient - (inverse of q) mod p (integer)
10. otherPrimeInfos - OtherPrimeInfos, which is another sequence of integers (optional, only required for version 1)

We used Lapo Luchini's ASN.1 decoder (https://lapo.it/asn1js/) on the DER setting, which decoded the file from base64. The structure of the file is as follows, with our notes:

SEQUENCE (9 elem)

Describes the data structure of the private key

Offset: 0

Length: 4+1764

(constructed)

Value: (9 elem)

1. INTEGER 0
    a. Version number - this is version 1
    b. Value: 0
    c. Offset: 4
    d. DER encoding: `02 01 00`
2. INTEGER (3072 bit)
    a. Modulus (n)
    b. Value: 486324801080346273921334679298060279973955007632031241551 1…
    c. Offset: 7
    d. DER encoding: `02 82 01 81`
3. INTEGER 65537
    a. publicExponent
    b. Value: 65537
    c. Offset: 396
    d. DER encoding: `02 03`
4. INTEGER (3072 bit)
    a. privateExponent
    b. Value: 460753364113093069472963235611881240274392181193531162841 6…
    c. Offset: 401
    d. DER encoding: `02 82 01 81`
5. INTEGER (1536 bit)
    a. prime1
    b. Value: 231115035060062997455182269650477242957433188832591929090 0…
    c. Offset: 790
    d. DER encoding: `02 81 C1`
6. INTEGER (1536 bit)
    a. prime2
    b. Value: 210425427732972220789107540510290689250966389632571539107 3…
    c. Offset: 986
    d. DER encoding: `02 81 C1`
7. INTEGER (1535 bit)
    a. exponent1
    b. Value: 120193077802647773658619974922520345327344788058367498041 9…

    c. Offset: 1182
    d. DER encoding: `02 81 C0`
8. INTEGER (1535 bit)
    a. exponent2
    b. Value: 119348209015705790911270219697392544824262197980260260313 4…
    c. Offset: 1377
    d. DER encoding: `02 81 C0`
9. INTEGER (1536 bit)
    a. coefficient
    b. Value: 171096582957620761803868817894416647931456238184308386561 4…
    c. Offset: 1572
    d. DER encoding: `02 81 C1`

Note: Bytes 3 on of the DER encoding is the length of the integer (ex. 0xC1 = 193 bytes = 1544 bits = metadata + data).

Public Key

Items in public key file:

1. modulus - the n in the RSA algorithm (integer)
2. publicExponent - the e in RSA (integer)

The modulus and the publicExponent are all that is needed to decrypt any messages encrypted in RSA with the secret key. In the in-class lab exercise, these two integers are all that we needed to swap to use RSA.

The public key is encoded in base64, with the addition of "ssh-rsa" at the beginning of the key and a user@host phrase at the end (kali@kali). The decoded key is as follows, with our labels added:

DER encoding for "ssh-rsa" (which is 7 bytes): `00 00 00 07`

"ssh-rsa": `73 73 68 2d 72 73 61`

DER encoding for public exponent (3 bytes): `00 00 00 03`

Public Exponent: `01 00 01`

DER encoding for modulus (0x181 bytes = 385 bytes): `00 00 01 81`

Modulus (n): `00 d6 4c 7d 00 18 17 6f 75 46 15 72 1c 61 1c cf 0b 81 66 5e 5f 3e e2 00 9a 43 7e b2 ac 48 29 f6 40 31 ab c5 3d 65 8f 82 0f a9 1c ce a7 2a a3 b3 b4 be dc de e5 5e 69 83 e3 1b 9d cf d3 5d 3f 4b d0 6a c9 b3 cf bd 85 96 17 bc 60 86 8a 6e 47 58 66 b9 3b 16 40 11 c1 43 7e bd d8 19 ba 94 72 34 e6 59 09 b1 06 6a 0b c8 c7 66 25 fe ec 16 f4 9f a5 14 f9 2e af 3c 54 5b d7 39 c2 f0 02 ba 64 15 1d c6 36 38 30 50 f7 c6 f5 d0 45 fa 6c 20 32 a2 0e 33 4d 7c 18 d0 90 b9 a7 e2 9a 97 d5 2a 2f e6 6e a9 82 2a d1 4e b6 8a 28 b9 11 00 9e b5 b9 30 1f 06 e7 e3 82 27 c0 40 d1 5d 36 09 4e 3f 75 0b a2 95 34 42 f8 b9 70 70 cc 5a 9e d1 ac d5 31 3c df 65 b3 39 c7 7b 1c b5 50 03 80 1c 6f 2b 1c 2c 80 34 7b 29 97 e5 22 04 2c 83 53 f9 ce 65 33 af 2c 05 8d 0c 28 f7 9b b0 2a 36 f1 aa f9 bb d4 98 40 d4 25 28 8b aa f3 7b 1c c5 cd b4 1a 64 c5 eb bc 87 f4 60 c7 dc 6e db e0 ee c5 0a 6d 79 6e 9e eb 2e 06 c5 00 aa 98 e0 af 81 63 40 c7 34 fe 64 17 3e d5 dc 78 f0 f6 cf da 23 ae 3c f4 7b 08 72 f9 a3 a4 1d 49 53 18 21 15 96 b0 cb 25 a1 0b af 67 5c da 95 b9 71 be 78 72 09 d7 dd b7 8c 1e c0 f0 39 63 20 c2 d2 74 f5 59 18 c5 78 be 58 bc df 28 d6 b5 a0 29 c2 06 94 25 6f ed 40 34 86 0e 1b 43`

Sanity Check

Using the values in the private key, we verified that:

- n = p*q
- With lambda(n) = lcm(p-1, q-1),
    - gcd(e, lambda(n)) = 1
    - (e*d) mod lambda(n) = 1
- The modulus n in the public key is the same as in the private key.

See ssh_testing.py for this code.