File: file-formats.pdf
File owner: Lysander Miller

SSH Key File Formats

Here are the contents of id_rsa_homework:

-----BEGIN RSA PRIVATE KEY-----
MIIG5AIBAAKCAYEA9MsrEtobf24LdKJny3qEzG5MsapMhMQa9YYVW8FuZ2n3rV
SA
ApI3PfXrcbeYZ46oDyTOJTARAEe5uhViVm5mk6ItOQPNz6wncXZByO26aJnQYp85
bianvwt2zohZn5k5HYzgevJgdXAhk4i+zPNpElry+f4McVwQkB0WZaqSPRpoET3L
We6sLSO6pqMb/9mjnUMQyrnp2y1gM/wbUCrxVre5d7gq8eHCVWQSeQMX2FP+gtEj
s0oLtKjjucaLPfZxmZKFMtJhvxILF2EMUHxFjwPE7J5gyAbPw4OkCH4VX//utAq5
gBGGZVEPUJUhuPImTeM6y+S9ZrotFDjIqzqFo0GyErYgJ4qAggT8Qx5is/VYOKhV
wD0zvgjFRxZZVRIAup5FY6OAAUU8k/p38VVI8UXF+GzOYJOACREvFUXBm18lyl
mV
+eBcE+J89zzDduMTMenrZx8EVi7zd659o910pUS1CIJt095x1bw3KXBLt2gsuBWj
V5PhrDSD3t6xq1H5AgMBAAECggGAXhrEgkjKjWBWkw3j7Ps6lgRJA3u+UsO4wW
48
Q7vGn4bDKiTz2Qkwp2ckjeDQe+8BeGDjxrQFtR/drEWQOj3SvSp5TBPESPObbijR
/VY2dQO8ck1XnJrLncvgbmFIYuxaYpvHqDwJDNyAa+EDyHJCkEXaZnRdgC6uR1iH
Raoe8dJ28znYNMhl9CxqEqldmCuolEo16Fk0J4f4FcPzFAyOHpIegIGSMM5jEMSZ
vV+NCSKxEsUnfkMW4OqUMNS8rqNQJApPu3arCiMMeDxNNSWa51fQiYFOVDNl
3zbO
1VrPvd5v9zlfIrkHvlBMuqRNYrXWwodDT9voO72hKZ3JUH0QBKqUhFcyiYr51tuN
4vWQ8he0y9zsevcdl77yFvbmOLKiTOz6xWYIiQGPD01isrji6qzAn9cUhvL+p9mN
0QXyCs2XEkra0HH3ax9DnRfpj93ye54jEDRx+AcWiSGpC+dMVUgOXmfEFObyCv
Wc
ms8EDU5iqz998pODUCTt0dJ1u/chAoHBAPticoaahWTdcryqKb3qDu5VnsiQfrGY
HkA9WrocpSxT1QURTQpKIqt6iYjscAq8NZYNwBVNBnQMl6gYlXYcqaabvIHTCiE
u
mxfxuLYaef6mpbaHFApjeG/ptT1YCrMJDBByHmOhgJ4mmjTgZbOd5qgfbtbkdMQi
UFiZI5xLf1Kov+cjlfhq8+ii2g1DVETkps6YHYYkXXKL1Be6OPHlWM4GCDRlSUUl
xtI9Ii+F9Zh1W3l5rFTfwLXpSa8YsBBXDQKBwQD5Sb4IWv4uSb6WfqiuDxfpOTxr
xfGjom7pD+IRNs+wfXEnc541RjIqoFUuAwv2UnU3O/ItOMnKUcIjmy5eNFnYfSPZ
zm3+F73O0a0eth3AvgOJpum3RvVWPqpd07b99uEJsdCv+G0tkcedD1yEmtB/frV2
CSYnnTVZxZkc9etg1rtnrCSC/CpblqDwZqj5TLOKbwVQiH/yZpjWxtNsUjcl6JK1
sKMZAREX8bRCFW5nG8mRA7fovLnpQ8opl7dx650CgcBQ1b1iDzopzxPgGw/FJAid
Fycx81TqIrJHkfMkuaVbdbGgKYoObvxrC8JCJ1V3/kF0+QL1Volms83onc8h5eGf

06BwVr6BIQ16S55L4IOuIURUR2doV8gYpJxFF5SJMbWRbEDdZMeJE3yu2CGb+oB
7
O3BW5auujiIr+0J6NcTBfcYHu2e0NCAuhH99mFL2vFfvQvbrdbIe7VvMLXO2RTuj
NElbH97Jv3YJkOL3SSpKViKOE1QZDsJXPKcXCOdEudkCgcEAzsT6qYWH5S7ntX5
T
PSRpydneounpdrepVQkGkw6qkBJMQ9Pjev7BZ5fbbzG5v1M/xFOlmRrMTVMpO01g
1WSIzUdm8CcIFsTse9pwxNN7tD4nQwq+OnXR0vphZzfPRbF7kQX7OapOLIkJT93I
+HhMziN0MfZ+vkboVJDQYjQcSxNxGBmxoy+zlopG7X/JUhbrqLxTZSwDLDUrOqJq
BSPgeEIDRk6/yWKYIgsqX9HU5BMpSm4SIio/7hp7Wapsz0IFAoHBALIXCZEhGWo
W
Jbv/qwNGQCiZTAkWDNoD6FVPiJeHyH/yFhvJctJlZ4V70E5Z+IBP94bSZXxobr/4
GHBodUvO+d0jZWAsp897L6Lmsq+iKREp6X1Lrlh6Wt4tsEeUyi3zUiyqaLKS6zkw
GXoMeVwG46Gs4GaW36sLd9KwLtu0dcPxri9BZEdRTpTrh5HXoVarohjAWd0YpxLp
crJdpvKwrBGYinVY11XVzza7/7k9O+Bcjb7cjUShb7B7vTILrMrYWw==
-----END RSA PRIVATE KEY-----

Here are the contents of id_rsa_homework.pub:

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQD0yysS2ht/bgt0omfLeoTMbkyxqkyExB
r1hhVbwW5nafetVIACkjc99etxt5hnjqgPJM4lMBEAR7m6FWJWbmaToi05A83PrCdxd
kHI7bpomdBi
nzluJqe/C3bOiFmfmTkdjOB68mB1cCGTiL7M82kSWvL5/gxxXBCQHRZlqpI9GmgRP
ctZ7qwtI7qmoxv/2aOdQxDKuenbLWAz/BtQKvFWt7l3uCrx4cJVZBJ5AxfYU/6C0SOz
Sgu0qOO5xos99nGZ
koUy0mG/EgsXYQxQfEWPA8TsnmDIBs/Dg6QIfhVf/+60CrmAEYZlUQ9QlSG48iZN
4zrL5L1mui0UOMirOoWjQbIStiAnioCCBPxDHmKz9Vg4qFXAPTO+CMVHFllVEgC
6nkVjo4ABRTyT+nfxVUjx
RcX4bM5gk4AJES8VRcGbXyXKWZX54FwT4nz3PMN24xMx6etnHwRWLvN3rn2j3
XSlRLUIgm3T3nHVvDcpcEu3aCy4FaNXk+GsNIPe3rGrUfk=
millerr2@LAPTOP-009N41E2

Private Key:

In my private key file (id_rsa_homework), I expect to see the following ASN.1 type RSAPrivateKey (Note: the following info is taken from https://datatracker.ietf.org/doc/html/rfc8017#appendix-A.1.2):

```
RSAPrivateKey ::= SEQUENCE {
    version           Version,
```

```
    modulus           INTEGER,  -- n
    publicExponent    INTEGER,  -- e
    privateExponent   INTEGER,  -- d
    prime1            INTEGER,  -- p
    prime2            INTEGER,  -- q
    exponent1         INTEGER,  -- d mod (p-1)
    exponent2         INTEGER,  -- d mod (q-1)
    coefficient       INTEGER,  -- (inverse of q) mod p
    otherPrimeInfos   OtherPrimeInfos OPTIONAL
}
```

The fields of the RSAPrivateKey can be described as follows:

- Version: The version number.
- Modulus: The RSA modulus n.
- PublicExponent: The RSA public exponent e.
- privateExponent: the RSA private exponent d.
- Prime1: the prime factor p of n.
- Prime2: the prime factor q of n.
- Exponent1: d mod (p-1)
- Exponent2: d mod (q-1)
- Coefficient: (inverse of q) mod p.
- otherPrimeInfos: the information for the additional primes r_3, ..., r_u, in order. It will be omitted if version is 0.

Now, let's go ahead and decode my private key file. To do so, I copied and pasted the contents of id_rsa_homework (from after -----BEGIN RSA PRIVATE KEY----- and before -----END RSA PRIVATE KEY-----) into https://holtstrom.com/michael/tools/asn1decoder.php. This gave me the following output:

```
SEQUENCE {
  INTEGER 0x00 (0 decimal)
  INTEGER
0x00f4cb2b12da1b7f6e0b74a267cb7a84cc6e4cb1aa4c84c41af586155bc16e6769f7ad548
00292373df5eb71b798678ea80f24ce2530110047b9ba1562566e6693a22d3903cdcfac277
17641c8edba6899d0629f396e26a7bf0b76ce88599f99391d8ce07af2607570219388beccf3
69125af2f9fe0c715c10901d1665aa923d1a68113dcb59eeac2d23baa6a31bffd9a39d4310c
ab9e9db2d6033fc1b502af156b7b977b82af1e1c2556412790317d853fe82d123b34a0bb4a
8e3b9c68b3df67199928532d261bf120b17610c507c458f03c4ec9e60c806cfc383a4087e1
55fffeeb40ab980118665510f509521b8f2264de33acbe4bd66ba2d1438c8ab3a85a341b212
```

b620278a808204fc431e62b3f55838a855c03d33be08c5471659551200ba9e4563a380014
53c93fa77f15548f145c5f86cce60938009112f1545c19b5f25ca5995f9e05c13e27cf73cc37
6e31331e9eb671f04562ef377ae7da3dd74a544b508826dd3de71d5bc3729704bb7682cb8
15a35793e1ac3483dedeb1ab51f9
INTEGER 0x010001 (65537 decimal)
INTEGER
0x5e1ac48248ca8d6056930de3ecfb3a960449037bbe52c3b8c16e3c43bbc69f86c32a24f3
d90930a767248de0d07bef017860e3c6b405b51fddac45903a3dd2bd2a794c13c448f39b6e
28d1fd56367503bc724d579c9acb9dcbe06e614862ec5a629bc7a83c090cdc806be103c872
429045da66745d802eae47588745aa1ef1d276f339d834c865f42c6a12a95d982ba8944a35
e859342787f815c3f3140c8e1e921e80819230ce6310c499bd5f8d0922b112c5277e4316e0
ea9430d4bcaea350240a4fbb76ab0a230c783c4d35259ae757d089814e543365df36ced55a
cfbdde6ff7395f22b907be504cbaa44d62b5d6c287434fdbe83bbda1299dc9507d1004aa948
45732898af9d6db8de2f590f217b4cbdcec7af71d97bef216f6e638b2a24cecfac5660889018
f0f4d62b2b8e2eaacc09fd71486f2fea7d98dd105f20acd97124adad071f76b1f439d17e98fd
df27b9e23103471f807168921a90be74c55480e5e67c414e6f20af59c9acf040d4e62ab3f7d
f293835024edd1d275bbf721
INTEGER
0x00fb6272869a8564dd72bcaa29bdea0eee559ec8907eb1981e403d5aba1ca52c53d50511
4d0a4a22ab7a8988ec700abc35960dc0154d06740c97a81895761ca9a69bbc81d30a212e9
b17f1b8b61a79fea6a5b687140a63786fe9b53d580ab3090c10721e63a1809e269a34e065b
39de6a81f6ed6e474c422505899239c4b7f52a8bfe72395f86af3e8a2da0d435444e4a6ce98
1d86245d728bd417ba38f1e558ce06083465494525c6d23d222f85f598755b7979ac54dfc0
b5e949af18b010570d
INTEGER
0x00f949be085afe2e49be967ea8ae0f17e9393c6bc5f1a3a26ee90fe21136cfb07d7127739e
3546322aa0552e030bf65275373bf22d38c9ca51c2239b2e5e3459d87d23d9ce6dfe17bdce
d1ad1eb61dc0be0389a6e9b746f5563eaa5dd3b6fdf6e109b1d0aff86d2d91c79d0f5c849ad
07f7eb5760926279d3559c5991cf5eb60d6bb67ac2482fc2a5b96a0f066a8f94cb38a6f0550
887ff26698d6c6d36c523725e892b5b0a319011117f1b442156e671bc99103b7e8bcb9e943
ca2997b771eb9d
INTEGER
0x50d5bd620f3a29cf13e01b0fc524089d172731f354ea22b24791f324b9a55b75b1a0298a
0e6efc6b0bc242275577fe4174f902f5568966b3cde89dcf21e5e19fd3a07056be81210d7a4
b9e4be083ae21445447676857c818a49c4517948931b5916c40dd64c789137caed8219bfa
807b3b7056e5abae8e222bfb427a35c4c17dc607bb67b434202e847f7d9852f6bc57ef42f6e
b75b21eed5bcc2d73b6453ba334495b1fdec9bf760990e2f7492a4a56228e1354190ec2573
ca71708e744b9d9
INTEGER
0x00cec4faa98587e52ee7b57e533d2469c9d9dea2e9e976b7a9550906930eaa90124c43d3

e37afec16797db6f31b9bf533fc453a5991acc4d53293b4d60d56488cd4766f0270816c4ec7
bda70c4d37bb43e27430abe3a75d1d2fa616737cf45b17b9105fb39aa4e2c89094fddc8f878
4cce237431f67ebe46e85490d062341c4b13711819b1a32fb3968a46ed7fc95216eba8bc53
652c032c352b3aa26a0523e0784203464ebfc96298220b2a5fd1d4e413294a6e12222a3fee
1a7b59aa6ccf4205
INTEGER
0x00b217099121196a1625bbffab03464028994c09160cda03e8554f889787c87ff2161bc9
72d26567857bd04e59f8804ff786d2657c686ebff8187068754bcef9dd2365602ca7cf7b2fa
2e6b2afa2291129e97d4bae587a5ade2db04794ca2df3522caa68b292eb3930197a0c795c0
6e3a1ace06696dfab0b77d2b02edbb475c3f1ae2f416447514e94eb8791d7a156aba218c05
9dd18a712e972b25da6f2b0ac11988a7558d755d5cf36bbffb93d3be05c8dbedc8d44a16fb0
7bbd320baccad85b
}

Now, let's talk about what each of these integers mean:

1. The first integer (in hexadecimal)
   0x00
   is the RSA version number. Because it's version 0, the otherPrimeInfos field will be omitted.

   The following bytes from the decoded base64 data represent the RSA version number:

   02 01 00

   The offset of the RSA version number within the bytes decoded from the base64 was 4. If we consider the DER encoding of the integer at that offset, we can observe that:

   Identifier octet:
   - 02 in binary is 00000010
   - Tag class = Universal
   - P/C = Primitive (P)
   - Tag type = INTEGER

   Length octet:
   - 01 in binary is 00000001
   - Form: Definite, short
   - Length: 1

   Contents octets:
   - There is one content octet

- The content is the RSA version number

2. The second integer (in hexadecimal)

   0x00f4cb2b12da1b7f6e0b74a267cb7a84cc6e4cb1aa4c84c41af586155bc16e6769f
   7ad54800292373df5eb71b798678ea80f24ce2530110047b9ba1562566e6693a22d
   3903cdcfac27717641c8edba6899d0629f396e26a7bf0b76ce88599f99391d8ce07af
   2607570219388beccf369125af2f9fe0c715c10901d1665aa923d1a68113dcb59eeac
   2d23baa6a31bffd9a39d4310cab9e9db2d6033fc1b502af156b7b977b82af1e1c2556
   412790317d853fe82d123b34a0bb4a8e3b9c68b3df67199928532d261bf120b1761
   0c507c458f03c4ec9e60c806cfc383a4087e155fffeeb40ab980118665510f509521b
   8f2264de33acbe4bd66ba2d1438c8ab3a85a341b212b620278a808204fc431e62b3f
   55838a855c03d33be08c5471659551200ba9e4563a38001453c93fa77f15548f145c
   5f86cce60938009112f1545c19b5f25ca5995f9e05c13e27cf73cc376e31331e9eb67
   1f04562ef377ae7da3dd74a544b508826dd3de71d5bc3729704bb7682cb815a3579
   3e1ac3483dedeb1ab51f9

   Is the modulus – the RSA modulus n.

   The following bytes from the decoded base64 data represent the RSA modulus n (excluding the … skipping 288 bytes … piece):

   02 82 01 81 00 F4 CB 2B 12 DA 1B 7F 6E 0B 74 A2 67 CB 7A 84 CC 6E 4C B1 AA 4C 84 C4 1A F5 86 15 5B C1 6E 67 69 F7 AD 54 80 02 92 37 3D F5 EB 71 B7 98 67 8E A8 0F 24 CE 25 30 11 00 47 B9 BA 15 62 56 6E 66 93 A2 2D 39 03
   … skipping 288 bytes …
   6D D3 DE 71 D5 BC 37 29 70 4B B7 68 2C B8 15 A3 57 93 E1 AC 34 83 DE DE B1 AB 51 F9

   The offset of the RSA modulus n within the bytes decoded from the base64 was 7. If we consider the DER encoding of the integer at that offset, we can observe that:

   Identifier octet:
   - Hex 02 in binary is 00000010
   - Tag class = Universal
   - P/C = Primitive (P)
   - Tag type = INTEGER

   Length octets:
   - Hex 82 in binary is 10000010

- Form: Definite, long
- Length: 2 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex 01 in binary is 00000001
    - Length octet 2 was hex 81
        - Hex 81 in binary is 10000001
    - In binary, the number of content octets used to store the RSA modulus n is 0000000110000001. This is equal to 385 in decimal.

Contents octets:

- There are 385 content octets
- The content is the RSA modulus n

3. The third integer (in hexadecimal)

    0x010001

    Is the publicExponent – the RSA public exponent e.

    The following bytes from the decoded base64 data represent the RSA public exponent e:

    02 03 01 00 01

    The offset of the RSA public exponent e within the bytes decoded from the base64 was 396. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010
    - Tag class = Universal
    - P/C = Primitive (P)
    - Tag type = INTEGER

    Length octets:

    - Hex 03 in binary is 00000011
    - Form: Definite, short
    - Length: 3

    Contents octets:

    - There are 3 content octets

- The content is the RSA public exponent e

4. The fourth integer (in hexadecimal)

   0x5e1ac48248ca8d6056930de3ecfb3a960449037bbe52c3b8c16e3c43bbc69f86c3
   2a24f3d90930a767248de0d07bef017860e3c6b405b51fddac45903a3dd2bd2a794c
   13c448f39b6e28d1fd56367503bc724d579c9acb9dcbe06e614862ec5a629bc7a83c
   090cdc806be103c872429045da66745d802eae47588745aa1ef1d276f339d834c865
   f42c6a12a95d982ba8944a35e859342787f815c3f3140c8e1e921e80819230ce6310
   c499bd5f8d0922b112c5277e4316e0ea9430d4bcaea350240a4fbb76ab0a230c783c
   4d35259ae757d089814e543365df36ced55acfbdde6ff7395f22b907be504cbaa44d6
   2b5d6c287434fdbe83bbda1299dc9507d1004aa94845732898af9d6db8de2f590f21
   7b4cbdcec7af71d97bef216f6e638b2a24cecfac5660889018f0f4d62b2b8e2eaacc09
   fd71486f2fea7d98dd105f20acd97124adad071f76b1f439d17e98fddf27b9e231034
   71f807168921a90be74c55480e5e67c414e6f20af59c9acf040d4e62ab3f7df293835
   024edd1d275bbf721

   Is the privateExponent – the RSA private exponent d.

   The following bytes from the decoded base64 data represent the RSA private exponent d:

   02 82 01 80 5E 1A C4 82 48 CA 8D 60 56 93 0D E3 EC FB 3A 96 04 49 03 7B BE 52
   C3 B8 C1 6E 3C 43 BB C6 9F 86 C3 2A 24 F3 D9 09 30 A7 67 24 8D E0 D0 7B EF 01
   78 60 E3 C6 B4 05 B5 1F DD AC 45 90 3A 3D D2 BD 2A 79 4C 13 C4 48 F3 9B 6E 28
   D1 … skipping 288 bytes …
   9A CF 04 0D 4E 62 AB 3F 7D F2 93 83 50 24 ED D1 D2 75 BB F7 21

   The offset of the RSA private exponent d within the bytes decoded from the base64 was 401. If we consider the DER encoding of the integer at that offset, we can observe that:

   Identifier octet:
   - Hex 02 in binary is 00000010
   - Tag class = Universal
   - P/C = Primitive (P)
   - Tag type = INTEGER

   Length octets:
   - Hex 82 in binary is 10000010
   - Form: Definite, long
   - Length: 2 (indicates how many length octets there will be after this one)

    - Length octet 1:
        - Hex 01 in binary is 00000001
    - Length octet 2 was hex 80
        - Hex 80 in binary is 10000000
    - In binary, the number of content octets used to store the RSA private exponent d is 0000000110000000. This is equal to 384 in decimal.

Contents octets:

- There are 384 content octets
- The content is the RSA private exponent d

5. The fifth integer (in hexadecimal)

    0x00fb6272869a8564dd72bcaa29bdea0eee559ec8907eb1981e403d5aba1ca52c53
    d505114d0a4a22ab7a8988ec700abc35960dc0154d06740c97a81895761ca9a69bb
    c81d30a212e9b17f1b8b61a79fea6a5b687140a63786fe9b53d580ab3090c10721e6
    3a1809e269a34e065b39de6a81f6ed6e474c422505899239c4b7f52a8bfe72395f86
    af3e8a2da0d435444e4a6ce981d86245d728bd417ba38f1e558ce06083465494525c
    6d23d222f85f598755b7979ac54dfc0b5e949af18b010570d

    Is the Prime1 – the prime factor p of n.

    The following bytes from the decoded base64 data represent the prime factor p of n:

    02 81 C1 00 FB 62 72 86 9A 85 64 DD 72 BC AA 29 BD EA 0E EE 55 9E C8 90 7E B1
    98 1E 40 3D 5A BA 1C A5 2C 53 D5 05 11 4D 0A 4A 22 AB 7A 89 88 EC 70 0A BC
    35 96 0D C0 15 4D 06 74 0C 97 A8 18 95 76 1C A9 A6 9B BC 81 D3 0A 21 2E …
    skipping 96 bytes …
    C6 D2 3D 22 2F 85 F5 98 75 5B 79 79 AC 54 DF C0 B5 E9 49 AF 18 B0 10 57 0D

    The offset of the prime factor p of n within the bytes decoded from the base64 was 789. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010
    - Tag class = Universal
    - P/C = Primitive (P)
    - Tag type = INTEGER

    Length octets:

- Hex 81 in binary is 10000001
- Form: Definite, long
- Length: 1 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex C1 in binary is 11000001
    - In binary, the number of content octets used to store the prime factor p of n is 11000001. This is equal to 193 in decimal.

Contents octets:

- There are 193 content octets
- The content is the prime factor p of n

6. The sixth integer (in hexadecimal)

    0x00f949be085afe2e49be967ea8ae0f17e9393c6bc5f1a3a26ee90fe21136cfb07d71
    27739e3546322aa0552e030bf65275373bf22d38c9ca51c2239b2e5e3459d87d23d
    9ce6dfe17bdced1ad1eb61dc0be0389a6e9b746f5563eaa5dd3b6fdf6e109b1d0aff8
    6d2d91c79d0f5c849ad07f7eb5760926279d3559c5991cf5eb60d6bb67ac2482fc2a
    5b96a0f066a8f94cb38a6f0550887ff26698d6c6d36c523725e892b5b0a319011117f
    1b442156e671bc99103b7e8bcb9e943ca2997b771eb9d

    Is the Prime2 – the prime factor q of n.

    The following bytes from the decoded base64 data represent the prime factor q of n:

    02 81 C1 00 F9 49 BE 08 5A FE 2E 49 BE 96 7E A8 AE 0F 17 E9 39 3C 6B C5 F1 A3
    A2 6E E9 0F E2 11 36 CF B0 7D 71 27 73 9E 35 46 32 2A A0 55 2E 03 0B F6 52 75 37
    3B F2 2D 38 C9 CA 51 C2 23 9B 2E 5E 34 59 D8 7D 23 D9 … skipping 96 bytes …
    B0 A3 19 01 11 17 F1 B4 42 15 6E 67 1B C9 91 03 B7 E8 BC B9 E9 43 CA 29 97 B7 71
    EB 9D

    The offset of the prime factor q of n within the bytes decoded from the base64 was 985. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010
    - Tag class = Universal
    - P/C = Primitive (P)
    - Tag type = INTEGER

Length octets:

- Hex 81 in binary is 10000001
- Form: Definite, long
- Length: 1 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex C1 in binary is 11000001
    - In binary, the number of content octets used to store the prime factor q of n is 11000001. This is equal to 193 in decimal.

Contents octets:

- There are 193 content octets
- The content is the prime factor q of n

7. The seventh integer (in hexadecimal)

    0x50d5bd620f3a29cf13e01b0fc524089d172731f354ea22b24791f324b9a55b75b1
    a0298a0e6efc6b0bc242275577fe4174f902f5568966b3cde89dcf21e5e19fd3a0705
    6be81210d7a4b9e4be083ae21445447676857c818a49c4517948931b5916c40dd64
    c789137caed8219bfa807b3b7056e5abae8e222bfb427a35c4c17dc607bb67b43420
    2e847f7d9852f6bc57ef42f6eb75b21eed5bcc2d73b6453ba334495b1fdec9bf76099
    0e2f7492a4a56228e1354190ec2573ca71708e744b9d9

    Is the Exponent1 – d mod (p-1).

    The following bytes from the decoded base64 data represent d mod (p - 1):

    02 81 C0 50 D5 BD 62 0F 3A 29 CF 13 E0 1B 0F C5 24 08 9D 17 27 31 F3 54 EA 22
    B2 47 91 F3 24 B9 A5 5B 75 B1 A0 29 8A 0E 6E FC 6B 0B C2 42 27 55 77 FE 41 74
    F9 02 F5 56 89 66 B3 CD E8 9D CF 21 E5 E1 9F D3 A0 70 56 BE 81 21 0D 7A 4B 9E
    4B E0 83 AE 21 … skipping 96 bytes …
    22 8E 13 54 19 0E C2 57 3C A7 17 08 E7 44 B9 D9

    The offset of d mod (p - 1) within the bytes decoded from the base64 was 1181. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010
    - Tag class = Universal
    - P/C = Primitive (P)

- Tag type = INTEGER

Length octets:

- Hex 81 in binary is 10000001
- Form: Definite, long
- Length: 1 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex C0 in binary is 11000000
    - In binary, the number of content octets used to store the prime factor q of n is 11000000. This is equal to 192 in decimal.

Contents octets:

- There are 192 content octets
- The content is d mod (p - 1)

8. The eighth integer (in hexadecimal)

    0x00cec4faa98587e52ee7b57e533d2469c9d9dea2e9e976b7a9550906930eaa9012
    4c43d3e37afec16797db6f31b9bf533fc453a5991acc4d53293b4d60d56488cd4766
    f0270816c4ec7bda70c4d37bb43e27430abe3a75d1d2fa616737cf45b17b9105fb39
    aa4e2c89094fddc8f8784cce237431f67ebe46e85490d062341c4b13711819b1a32f
    b3968a46ed7fc95216eba8bc53652c032c352b3aa26a0523e0784203464ebfc96298
    220b2a5fd1d4e413294a6e12222a3fee1a7b59aa6ccf4205

    Is the Exponent2 – d mod (q-1).

    The following bytes from the decoded base64 data represent d mod (q - 1):

    02 81 C1 00 CE C4 FA A9 85 87 E5 2E E7 B5 7E 53 3D 24 69 C9 D9 DE A2 E9 E9 76
    B7 A9 55 09 06 93 0E AA 90 12 4C 43 D3 E3 7A FE C1 67 97 DB 6F 31 B9 BF 53 3F
    C4 53 A5 99 1A CC 4D 53 29 3B 4D 60 D5 64 88 CD 47 66 F0 27 08 16 C4 EC 7B DA
    70 C4 … skipping 96 bytes …
    D1 D4 E4 13 29 4A 6E 12 22 2A 3F EE 1A 7B 59 AA 6C CF 42 05

    The offset of d mod (q - 1) within the bytes decoded from the base64 was 1376. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010
    - Tag class = Universal

- P/C = Primitive (P)
- Tag type = INTEGER

Length octets:

- Hex 81 in binary is 10000001
- Form: Definite, long
- Length: 1 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex C1 in binary is 11000001
    - In binary, the number of content octets used to store d mod (q - 1) is 11000001. This is equal to 193 in decimal.

Contents octets:

- There are 193 content octets
- The content is d mod (q-1)

9. The ninth integer (in hexadecimal)

    0x00b217099121196a1625bbffab03464028994c09160cda03e8554f889787c87ff2161bc972d26567857bd04e59f8804ff786d2657c686ebff8187068754bcef9dd2365602ca7cf7b2fa2e6b2afa2291129e97d4bae587a5ade2db04794ca2df3522caa68b292eb3930197a0c795c06e3a1ace06696dfab0b77d2b02edbb475c3f1ae2f416447514e94eb8791d7a156aba218c059dd18a712e972b25da6f2b0ac11988a7558d755d5cf36bbffb93d3be05c8dbedc8d44a16fb07bbd320baccad85b

    Is the Coefficient – CRT coefficient q^(-1) mod p.

    The following bytes from the decoded base64 data represent q^(-1) mod p:

    02 81 C1 00 B2 17 09 91 21 19 6A 16 25 BB FF AB 03 46 40 28 99 4C 09 16 0C DA 03 E8 55 4F 88 97 87 C8 7F F2 16 1B C9 72 D2 65 67 85 7B D0 4E 59 F8 80 4F F7 86 D2 65 7C 68 6E BF F8 18 70 68 75 4B CE F9 DD 23 65 60 2C A7 CF 7B 2F … skipping 96 bytes …
    36 BB FF B9 3D 3B E0 5C 8D BE DC 8D 44 A1 6F B0 7B BD 32 0B AC CA D8 5B

    The offset of q^(-1) mod p within the bytes decoded from the base64 was 1572. If we consider the DER encoding of the integer at that offset, we can observe that:

    Identifier octet:

    - Hex 02 in binary is 00000010

- Tag class = Universal
- P/C = Primitive (P)
- Tag type = INTEGER

Length octets:

- Hex 81 in binary is 10000001
- Form: Definite, long
- Length: 1 (indicates how many length octets there will be after this one)
    - Length octet 1:
        - Hex C1 in binary is 11000001
    - In binary, the number of content octets used to store q^(-1) mod p is 11000001. This is equal to 193 in decimal.

Contents octets:

- There are 193 content octets
- The content is q^(-1) mod p

Public Key:

In my public key file (id_rsa_homework.pub), I expect to find:

- n — the RSA modulus, a positive integer
- e — the RSA public exponent, a positive integer

I got this information from PKCS #1: RSA Cryptography Specifications Version 2.2 section 3.1 (https://datatracker.ietf.org/doc/html/rfc8017#section-3.1).

I decoded the file into hexadecimal by running the following command (which I got from https://www.thedigitalcatonline.com/blog/2018/04/25/rsa-keys/):

```
cat id_rsa_homework.pub | cut -d " " -f2 | base64 -d | hexdump -ve '/1 "%02x "' -e '2/8 "\n"'
```

The output of this command was the following:

```
00 00 00 07 73 73 68 2d 72 73 61 00 00 00 03 01 00 01 00 00 01 81 00 f4 cb 2b
12 da 1b 7f 6e 0b 74 a2 67 cb 7a 84 cc 6e 4c b1 aa 4c 84 c4 1a f5 86 15 5b c1 6e
67 69 f7 ad 54 80 02 92 37 3d f5 eb 71 b7 98 67 8e a8 0f 24 ce 25 30 11 00 47 b9
ba 15 62 56 6e 66 93 a2 2d 39 03 cd cf ac 27 71 76 41 c8 ed ba 68 99 d0 62 9f 39
6e 26 a7 bf 0b 76 ce 88 59 9f 99 39 1d 8c e0 7a f2 60 75 70 21 93 88 be cc f3 69
12 5a f2 f9 fe 0c 71 5c 10 90 1d 16 65 aa 92 3d 1a 68 11 3d cb 59 ee ac 2d 23 ba
```

a6 a3 1b ff d9 a3 9d 43 10 ca b9 e9 db 2d 60 33 fc 1b 50 2a f1 56 b7 b9 77 b8 2a
f1 e1 c2 55 64 12 79 03 17 d8 53 fe 82 d1 23 b3 4a 0b b4 a8 e3 b9 c6 8b 3d f6 71
99 92 85 32 d2 61 bf 12 0b 17 61 0c 50 7c 45 8f 03 c4 ec 9e 60 c8 06 cf c3
83 a4 08 7e 15 5f ff ee b4 0a b9 80 11 86 65 51 0f 50 95 21 b8 f2 26 4d e3 3a cb
e4 bd 66 ba 2d 14 38 c8 ab 3a 85 a3 41 b2 12 b6 20 27 8a 80 82 04 fc 43 1e 62 b3
f5 58 38 a8 55 c0 3d 33 be 08 c5 47 16 59 55 12 00 ba 9e 45 63 a3 80 01 45 3c
93 fa 77 f1 55 48 f1 45 c5 f8 6c ce 60 93 80 09 11 2f 15 45 c1 9b 5f 25 ca 59 95
f9 e0 5c 13 e2 7c f7 3c c3 76 e3 13 31 e9 eb 67 1f 04 56 2e f3 77 ae 7d a3 dd 74
a5 44 b5 08 82 6d d3 de 71 d5 bc 37 29 70 4b b7 68 2c b8 15 a3 57 93 e1 ac 34
83 de de b1 ab 51 f9

Here is a bit of info about the above output (used https://www.thedigitalcatonline.com/blog/2018/04/25/rsa-keys/ as reference):

(4 bytes) 00 00 00 07 = 7
(7 bytes) 73 73 68 2d 72 73 61 = "ssh-rsa" (US-ASCII)
(4 bytes) 00 00 00 03 = 3
(3 bytes) 01 00 01 = 65537 (the RSA exponent)
(4 bytes) 00 00 01 81 = 385 (the number of bytes in the RSA modulus)
(385 bytes) 00 f4 cb .. e1 ac 34 = The RSA modulus

Sanity Check:

Let's check that the integers we've found in these two files work as we would expect from an RSA key pair.

First, n should be equal to p * q.

n from both of the files (id_rsa_homework and id_rsa_homework.pub) is equal to

0x00f4cb2b12da1b7f6e0b74a267cb7a84cc6e4cb1aa4c84c41af586155bc16e6769f7ad548
00292373df5eb71b798678ea80f24ce2530110047b9ba1562566e6693a22d3903cdcfac277
17641c8edba6899d0629f396e26a7bf0b76ce88599f99391d8ce07af2607570219388beccf3
69125af2f9fe0c715c10901d1665aa923d1a68113dcb59eeac2d23baa6a31bffd9a39d4310c
ab9e9db2d6033fc1b502af156b7b977b82af1e1c2556412790317d853fe82d123b34a0bb4a
8e3b9c68b3df67199928532d261bf120b17610c507c458f03c4ec9e60c806cfc383a4087e1
55fffeeb40ab980118665510f509521b8f2264de33acbe4bd66ba2d1438c8ab3a85a341b212
b620278a808204fc431e62b3f55838a855c03d33be08c5471659551200ba9e4563a380014
53c93fa77f15548f145c5f86cce60938009112f1545c19b5f25ca5995f9e05c13e27cf73cc37

6e31331e9eb671f04562ef377ae7da3dd74a544b508826dd3de71d5bc3729704bb7682cb8
15a35793e1ac3483dedeb1ab51f9

Using this website (https://onlinehextools.com/multiply-hex-numbers), I performed p * q and got n back. Thus, we know n == pq.

Now, we know that both of the following should be true:

1) e should be less than (p-1) * (q-1)

Using this website (https://onlinehextools.com/multiply-hex-numbers), I performed (p-1) * (q-1) and got the following:

0xf4cb2b12da1b7f6e0b74a267cb7a84cc6e4cb1aa4c84c41af586155bc16e
6769f7ad54800292373df5eb71b798678ea80f24ce2530110047b9ba15625
66e6693a22d3903cdcfac27717641c8edba6899d0629f396e26a7bf0b76ce8
8599f99391d8ce07af2607570219388beccf369125af2f9fe0c715c10901d16
65aa923d1a68113dcb59eeac2d23baa6a31bffd9a39d4310cab9e9db2d6033
fc1b502af156b7b977b82af1e1c2556412790317d853fe82d123b34a0bb4a8
e3b9c68b3df66fa4e654a3dcde2bead9c43839e4831eb774e9b847f072cc42
9a36673d2aa0832ea88785ce4081318f363099a5a5b7e37b809988895eed2
85bd3488504d6e9822428d1b714a80844279b41966e4dcaf490aef87a57fe9
2c5f7e065abea97a41213a07a56500992358cdac20ab5b91c91c9a7a1ab7f3
374d3fa56b2725c0456e7dfaa7b89db7e793dcfa901f992c638b1b8d8e2c89
07226f13c86892d7145a5e4b1561a1e6c8175b799eb8d60b7d4e7b7d707ad
983f19f7670d0770062e4a290f50

This is definitely bigger than 0x010001

2) e and (p-1) * (q-1)'s greatest common divisor should be 1

Using this website (https://onlinemathtools.com/find-all-divisors), I learned that 0x010001 (well, technically I used the decimal version – which was 65537) has two divisors: itself and one.

Then, I used this website (https://www.dcode.fr/big-numbers-division) to divide the decimal version of (p-1) * (q-1) by the decimal equivalent of e. This got me:

847657213939751943236242235499495333234981022362113888180708195141
459667734469856685504135319779588323299888829292380486431808504834

637634176366628762376423770265729461473246181631393126524484363660
382942597288103717246047822090475428521786502340566556449565645904
035133314040793303255317220528908825399700714190103640918266283972
211935020875122075992969887460948960285007543687876841556428289741
321242308530928321630907924664133638782976232968134607668100745777
666238742291306742059456371701700549152175179040725856619815074367
948620046208082205552820297361013960940336966165380101370702200124
294881520179263416548668665888533815309490612411196161639753991793
966101082286469428842224799934223483451625581902030543298742268823
516900389054477600331240443433439808445412746646120323439870216821
710674611159303218695178710985294083028873036133992746166449988627
108362423836631360442975207028199315091362667672064684305009 6.901
841707737613866975906739704289180157773471474129117902864031005386
270351099378976761218853472084 4

Which is not a whole number. Thus, e and (p-1) * (q-1)’s greatest common divisor is 1.

Finally, e*d mod ((p-1) * (q-1)) should be equal to one. Let’s test if this is the case.

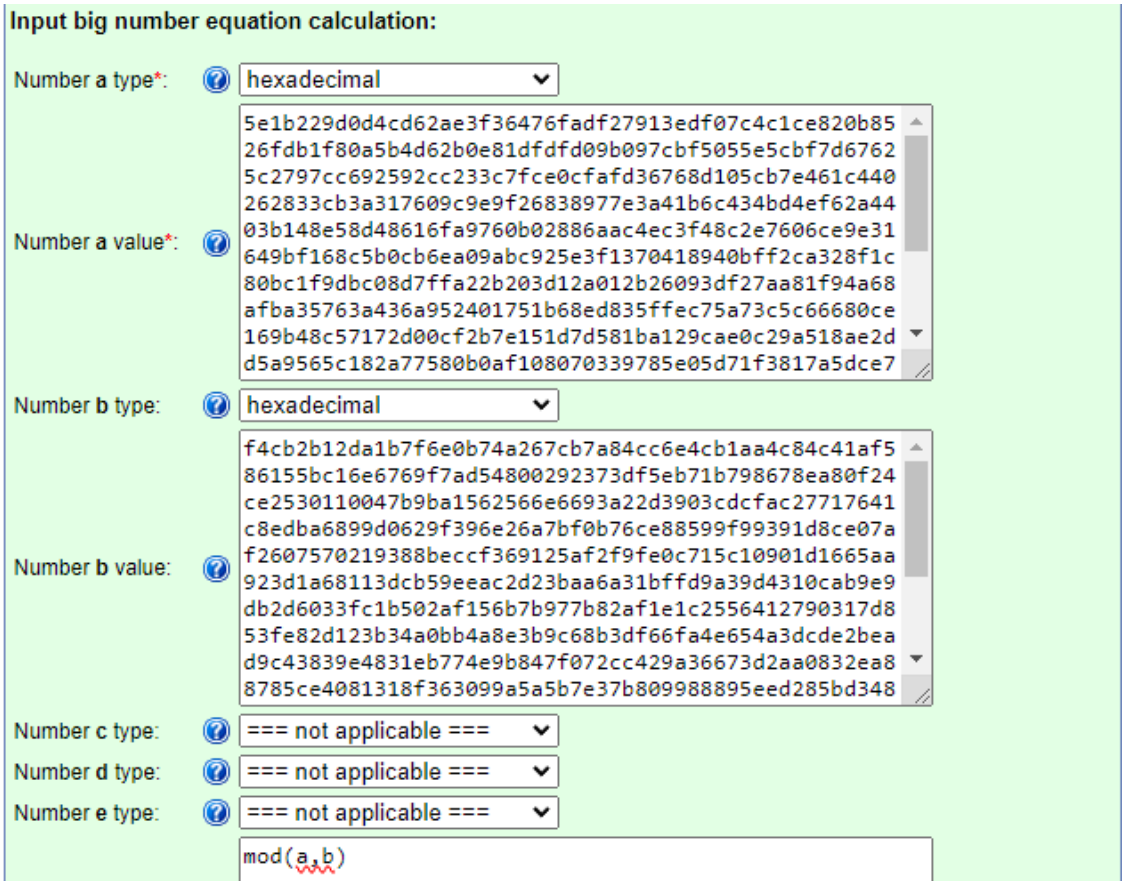
Using this website (https://onlinehextools.com/multiply-hex-numbers), I have calculated that e * d is:

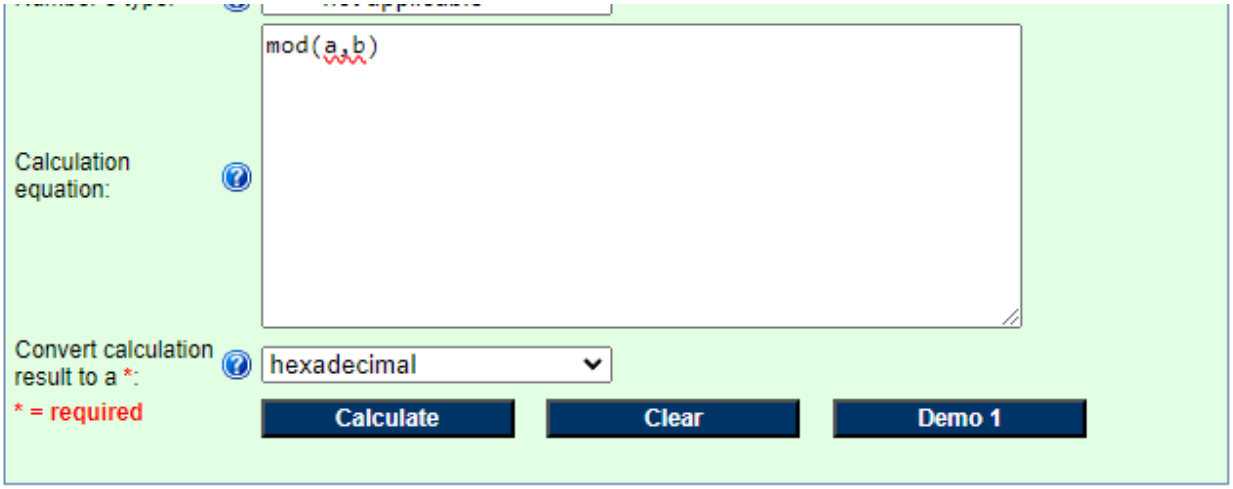
0x5e1b229d0d4cd62ae3f36476fadf27913edf07c4c1ce820b8526fdb1f80a5
b4d62b0e81dfdfd09b097cbf5055e5cbf7d67625c2797cc692592cc233c7fce
0cfafd36768d105cb7e461c440262833cb3a317609c9e9f26838977e3a41b6
c434bd4ef62a4403b148e58d48616fa9760b02886aac4ec3f48c2e7606ce9e
31649bf168c5b0cb6ea09abc925e3f1370418940bff2ca328f1c80bc1f9dbc0
8d7ffa22b203d12a012b26093df27aa81f94a68afba35763a436a952401751
b68ed835ffec75a73c5c66680ce169b48c57172d00cf2b7e151d7d581ba129
cae0c29a518ae2dd5a9565c182a77580b0af108070339785e05d71f3817a5d
ce73ef2ee466081ba992eebb6e0bd8361d564be8373e7a8a6e391b857e3988
edc89d5e8fd2f98daff8f47b260cdef0a1810dc72001b95a38fab4c97b45e07
859ad835aa93c310bfa1dfe1ed1b4cc7dd16aebc5b86a7c78259909eae5782
2c790e9038322d90583c9463a676227cab06f1e7a7906b9edc526ff9a22932
1175e3a83df6c0478e6cdcf721

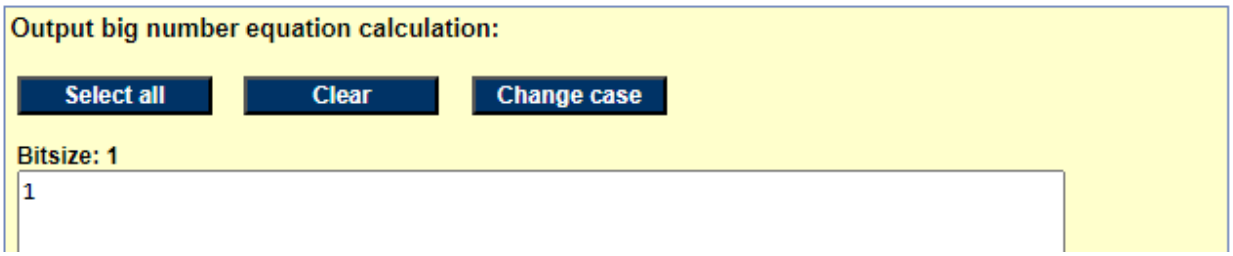
Using this website (https://onlinehextools.com/multiply-hex-numbers), I performed (p-1) * (q-1) and got the following:

0xf4cb2b12da1b7f6e0b74a267cb7a84cc6e4cb1aa4c84c41af586155bc16e
6769f7ad54800292373df5eb71b798678ea80f24ce2530110047b9ba15625
66e6693a22d3903cdcfac27717641c8edba6899d0629f396e26a7bf0b76ce8
8599f99391d8ce07af2607570219388beccf369125af2f9fe0c715c10901d16
65aa923d1a68113dcb59eeac2d23baa6a31bffd9a39d4310cab9e9db2d6033
fc1b502af156b7b977b82af1e1c2556412790317d853fe82d123b34a0bb4a8
e3b9c68b3df66fa4e654a3dcde2bead9c43839e4831eb774e9b847f072cc42
9a36673d2aa0832ea88785ce4081318f363099a5a5b7e37b809988895eed2
85bd3488504d6e9822428d1b714a80844279b41966e4dcaf490aef87a57fe9
2c5f7e065abea97a41213a07a56500992358cdac20ab5b91c91c9a7a1ab7f3
374d3fa56b2725c0456e7dfaa7b89db7e793dcfa901f992c638b1b8d8e2c89
07226f13c86892d7145a5e4b1561a1e6c8175b799eb8d60b7d4e7b7d707ad
983f19f7670d0770062e4a290f50

Finally, using this website (https://www.mobilefish.com/services/big_number_equation/big_number_equation.php#equation_output) I calculated e*d mod ((p-1) * (q-1)) and got one!







So, in conclusion, we can be confident that the integers we found in these two files work as we'd expect from an RSA key pair.