Names: Luisa Escosteguy, PJ Sangvong

File contents

Private key: id_rsa_homework

-----BEGIN RSA PRIVATE KEY-----
MIIG5AIBAAKCAYEA2LNRdHb+3/q0Lb0VMrvXs/AsPeeB5e+eo0AHIF0jZzHLMe2w
yAL59yyB0kIl0gNoLmaNfd3zjI3VCRE+TXU0Xfbb4LwgTS7WKe+1ZPAkrjBrgLUW
0garVFuB/7jSatiigyV+GqBXFIhx76sNVzFHUk4ClmL1sB+8MY+vfImgbUu2oMQm
jcpI1VmPqWcZEFfcpr5MAlftkbwiY8bYzYNN6fNljIA0+NTiW9133B+tohndOUzf
zhCwWpz9jU9ZlYB1pabtgZVAKiPDirEBWAkQXQlsTR+a60FZtzYuuPmH2STqnniw
PqxnONkx8RCLdKjo4XBjZ5CQ4gLryvTEvQRavxOJ8PMkdsQj67amC0zRfLbF094/
th3AbHUrUPpRQobzF9fpzipJwfy57rUcUVYaaVAHlEf3aZcCcDr+KAszTOrUvMti
Q0XK2vJSIbeJGa53JSLaa9ZN88W/Zs+dpidoK2pZWaMzT4vKnmwZJ4Ex34M4mQs7
civzfhLvjACE5Un/AgMBAAECggGBAJtHOxn6Xr4+VjfkIrl88p7kfb9KEcGjB4ix
S6n8M28xtgmr6Z4Yy+c7BDeW4KTxfgipwb+seGWCCGJ78antTz35sysBgx/rbNkW
BGQ01APhfFb47EZJMyG6hJeqZ1TasPDKv+byeBOlhgcYCfItxI6MibwbYzP2OZNG
A5h4wfUvaMkgZQP7LF980r2vu70DSIBPIgYX3YstRXM7vITfDsiSeXYVsS+Q1DbB
pq6cIH5kjCoQFNsDCGuVzPYiIu5G4jSSRLW40byRKMZKqhDVKzRhQvCOBj3Ry13c
73xGyT+ogN0f7mIs/ziaSsCd86aCRNkVW2MrDFmlFuheCYOS2s4yimTXUvgMKn2W
/BHW7tgHRHqfvDxJKzAhD2RgwdryGGHi4iO1Y3bxabF2x9E6t9K9lU/ITVW8Lcq9
7VoghtIAdJONo+6ubm+d4lsR0oo3l1ctl6OUFzQwgXYXegGtQ8ozKeYmj8thNQi6
ocRpMBS7QoLEm+Gx86dl++kJoI2iWQKBwQD2XK/snfKyjBQnxOKB+g0M6u/it/pD
MF/kjBvlyqQov6vNBRfYquxwn0qUXsSu+rKhZ45nltUKHUGxunqpBXyXqLukL9Rj
Kc2IcN+V33YYA3d3+xnsKv9NuOaIUgv7mO9Z6fIcW81x7YJ6TiNFsuAN+az30ciw
uDwo2gwGe9NRSe7O8rRo7Hkke13DJrSdXjEvVRDLrWeavTNZVKhsQjh9akCg5HW+
wfovkkNFMyu49FmDkoKM3pDTjkF2ZC1mDmsCgcEA4S2SChiNoJIcqrP92PJSh1ra
MMHU9Rq1/9JQON2JTE91ggmj9GdFKj5o+plt6bg+8W49l3zKt1x2Wh19mcvoG5ov
hE7flhib8rdgnMfDfJv+UU9/we1ZekttwO6Uo6FCESznLeMNcnrVVjtjTYIys1uW
QfXzP3dclYPo//QIuHdy0vyL1blkkC2fly1GQ/3+SMoznWy3LmwCL2pJhlOwCifl
Hzup2/1MvXYApkOZMJj7eWkCCj8y+zuQqHNHdS+9AoHAf+G/BEE88QebksU1mqiU
y6bMXNCJXQUoUbeU56RXsDtGT8ccCME4uIFHMls7F2VVAkmB9y2panWjW0FkWIZb
lVHmDSUyx0K0yOVmeiKj53UtN5+lZh0vp0WajRqh07kgTXkNz0CsFT5no1saHhzr
BvYang1Mcsc/mykMYvU1zRVmnBAKLAjMHW7YTeuyh5mUSHCMpl1d0Uny9KToREHf
luJnp9zPbfS9DKKMvsgAIyLbsFuF3t/NJ7PJMU5w7m//AoHBAIyvnhi0snfqEgMN
JcKP08RBnSTtVZI1uBQ3YGvG7etBkddHEqq2gt4b1BcOGT/3H0xUZiLpkvXwzPkM
h9a5MTFdqeZuKP9sLLL+o1wQ6zBFfiVrGBUfp9HiopXx+egp7k8w0Nc4jsRBvlx8
ClT4zZiF1mEru2ihuGwwMDkKTRCDgLgVIYBYrl3uQ1F+tfHxFRiBTLJEpEZasRrS
u1EzpD8UD9KDwmJx5apRuaInheR5EFUQqHeieMXCt2SncbPjuQKBwGDobWGr9+bN
tOiKCCqBp1+WEzldTNrZRaeOZt1e45zQsXKZ25RwuTJ0kQ5+J7uMlmpCWCwVXJhY
ZnKvXCWIybSeqPAr/gGPrjEF3XAhlnzmG5xmW2RySq520X1+Se80r0noT4MhDJTG
SrrMZTbtQyyqTamb3KHByKfqJ/BzlMvXBKNdj/VS+GiJ0OUznlthgrRhVa2XsgJn
71cJG79kvA5TXYWmigQEfCCWRQJteCYwJ3DXyjWV7meHPPPf1gGhrw==
-----END RSA PRIVATE KEY-----

Public key: id_rsa_homework.pub

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDYs1F0dv7f+rQtvRUyu9ez8Cw954Hl756jQAcgXSNnMcsx7bDIAvn3LIHSQiXSA
2guZo193fOMjdUJET5NdTRd9tvgvCBNLtYp77Vk8CSuMGuAtRbSBqtUW4H/uNJq2KKDJX4aoFcUiHHvqw1XMUdSTgKWY
vWwH7wxj698iaBtS7agxCaNykjVWY+pZxkQV9ymvkwCV+2RvCJjxtjNg03p82WOUDT41OJb3XfcH62iGd05TN/OELBanP2
NT1mVgHWlpu2BlUAqI8OKsQFYCRBdCWxNH5rrQVm3Ni64+YfZJOqeeLA+rGc42THxElt0qOjhcGNnkJDiAuvK9MS9BFq/

E4nw8yR2xCPrtqYLTNF8tsXT3j+2HcBsdStQ+lFChvMX1+nOKknB/LnutRxRVhppUAeUR/dplwJwOv4oCzNM6tS8y2JDRcra
8llht4kZrnclltpr1k3zxb9mz52mJ2grallZozNPi8qebBkngTHfgziZCztyK/N+Eu+MAITlSf8=
luisaescosteguy@MacBook-Pro-de-Luisa.local

We needed a different format, so we used the command `ssh-keygen -e -f id_rsa_homework.pub -m PKCS8` to get the public key in PEM/PKCS#8 (more on that later):

-----BEGIN PUBLIC KEY-----
MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEA2LNRdHb+3/q0Lb0VMrvX
s/AsPeeB5e+eo0AHIF0jZzHLMe2wyAL59yyB0kIl0gNoLmaNfd3zjI3VCRE+TXU0
Xfbb4LwgTS7WKe+1ZPAkrjBrgLUW0garVFuB/7jSatiigyV+GqBXFIhx76sNVzFH
Uk4ClmL1sB+8MY+vfImgbUu2oMQmjcpI1VmPqWcZEFfcpr5MAlftkbwiY8bYzYNN
6fNljIA0+NTiW9133B+tohndOUzfzhCwWpz9jU9ZlYB1pabtgZVAKiPDirEBWAkQ
XQlsTR+a60FZtzYuuPmH2STqnniwPqxnONkx8RCLdKjo4XBjZ5CQ4gLryvTEvQRa
vxOJ8PMkdsQj67amC0zRfLbF094/th3AbHUrUPpRQobzF9fpzipJwfy57rUcUVYa
aVAHlEf3aZcCcDr+KAszTOrUvMtiQ0XK2vJSIbeJGa53JSLaa9ZN88W/Zs+dpido
K2pZWaMzT4vKnmwZJ4Ex34M4mQs7civzfhLvjACE5Un/AgMBAAE=
-----END PUBLIC KEY-----

Private key

From the RFC 8017 Appendix 1.2, we expect to find 8 or 9 (one is optional) integers and a version to be in the file in this order:

1. version
2. modulus
3. publicExponent
4. privateExponent
5. prime1
6. prime2
7. exponent1
8. exponent2
9. coefficient
10. [OPTIONAL] otherPrimeInfos.

Decoded the private key file with [Michael Holtstrom's ASN.1](#) Decoder

To decode the private key file, we copied the contents of id_rsa_homework to the decoder. We select auto-detect and press convert. We got the following output:

SEQUENCE {
  INTEGER 0x00 (0 decimal)
  INTEGER
0x00d8b3517476fedffab42dbd1532bbd7b3f02c3de781e5ef9ea34007205d236731cb31edb0c802f9f72c81d24225d203682e668d7dddf38c8dd509113e4
d75345df6dbe0bc204d2ed629efb564f024ae306b80b516d206ab545b81ffb8d26ad8a283257e1aa057148871efab0d573147524e029662f5b01fbc318faf
7c89a06d4bb6a0c4268dca48d5598fa967191057dca6be4c0257ed91bc2263c6d8cd834de9f3658e5034f8d4e25bdd77dc1fada219dd394cdfce10b05a9
cfd8d4f59958075a5a6ed8195402a23c38ab1015809105d096c4d1f9aeb4159b7362eb8f987d924ea9e78b03eac6738d931f1108b74a8e8e17063679090
e202ebcaf4c4bd045abf1389f0f32476c423ebb6a60b4cd17cb6c5d3de3fb61dc06c752b50fa514286f317d7e9ce2a49c1fcb9eeb51c51561a6950079447f
7699702703afe280b334cead4bccb624345cadaf25221b78919ae772522da6bd64df3c5bf66cf9da627682b6a5959a3334f8bca9e6c19278131df8338990
b3b722bf37e12ef8c0084e549ff
  INTEGER 0x010001 (65537 decimal)
  INTEGER
0x009b473b19fa5ebe3e5637e422b97cf29ee47dbf4a11c1a30788b14ba9fc336f31b609abe99e18cbe73b043796e0a4f17e08a9c1bfac78658208627bf1a

9ed4f3df9b32b01831feb6cd916046434d403e17c56f8ec46493321ba8497aa6754dab0f0cabfe6f27813a586071809f22dc48e8c89bc1b6333f639934603
9878c1f52f68c9206503fb2c5f7cd2bdafbbbd0348804f220617dd8b2d45733bbc84df0ec892797615b12f90d436c1a6ae9c207e648c2a1014db03086b95c
cf62222ee46e2349244b5b8d1bc9128c64aaa10d52b346142f08e063dd1cb5ddcef7c46c93fa880dd1fee622cff389a4ac09df3a68244d9155b632b0c59a5
16e85e098392dace328a64d752f80c2a7d96fc11d6eed807447a9fbc3c492b30210f6460c1daf21861e2e223b56376f169b176c7d13ab7d2bd954fc84d55
bc2dcabded5a2086d20074938da3eeae6e6f9de25b11d28a3723572d97a3941734308176177a01ad43ca3329e6268fcb613508baa1c4693014bb4282c4
9be1b1f3a765fbe909a08da259
  INTEGER
0x00f65cafec9df2b28c1427c4e281fa0d0ceaefe2b7fa43305fe48c1be5caa428bfabcd0517d8aaec709f4a945ec4aefab2a1678e6796d50a1d41b1ba7aa9
057c97a8bba42fd46329cd8870df95df7618037777fb19ec2aff4db8e688520bfb98ef59e9f21c5bcd71ed827a4e2345b2e00df9acf7d1c8b0b83c28da0c06
7bd35149eecef2b468ec79247b5dc326b49d5e312f5510cbad679abd335954a86c42387d6a40a0e475bec1fa2f924345332bb8f4598392828cde90d38e4
176642d660e6b
  INTEGER
0x00e12d920a188da0921caab3fdd8f252875ada30c1d4f51ab5ffd25038dd894c4f758209a3f467452a3e68fa996de9b83ef16e3d977ccab75c765a1d7d9
9cbe81b9a2f844edf96189bf2b7609cc7c37c9bfe514f7fc1ed597a4b6dc0ee94a3a142112ce72de30d727ad5563b634d8232b35b9641f5f33f775c2183e8f
ff408b87772d2fc8bd5b964902d9f972d4643fdfe48ca339d6cb72e6c022f6a498653b00a27c81f3ba9dbfd4cbd7600a643993098fb7969020a3f32fb3b90a
87347752fbd
  INTEGER
0x7fe1bf04413cf1079b92c5359aa894cba6cc5cd0895d052851b794e7a457b03b464fc71c08c138b88147325b3b176555024981f72da96a75a35b41645
8865b9551e60d2532c742b4c8e5667a22a3e7752d379fa5661d2fa7459a8d1aa1d3b9204d790dcf40ac153e67a35b1a1e1ceb06f61a9e0d4c72c73f9b29
0c62f535cd15669c100a2c08cc1d6ed84debb287999448708ca65d5dd149f2f4a4e84441df96e267a7dccf6df4bd0ca28cbec8002322dbb05b85dedfcd27b
3c9314e70ee6fff
  INTEGER
0x008caf9e18b4b277ea12030d25c28fd3c4419d24ed559235b81437606bc6edeb4191d74712aab682de1bd4170e193ff71f4c546622e992f5f0ccf90c87d
6b931315da9e66e28ff6c2cb2fea35c10eb30457e256b18151fa7d1e2a295f1f9e829ee4f30d0d7388ec441be5c7c0a54f8cd9885d6612bbb68a1b86c3030
390a4d108380b815958058ae5dee43517eb5f1f11518814cb244a4465ab11ad2bb5133a43f140fd283c26271e5aa51b9a22785e479105510a877a278c5
c2b764a771b3e3b9
  INTEGER
0x60e86d61abf7e6cdb4e88a082a81a75f9613395d4cdad945a78e66dd5ee39cd0b17299db9470b93274910e7e27bb8c966a42582c155c98586672af5c
2588c9b49ea8f02bfe018fae3105dd7021967ce61b9c665b64724aae76d17d7e49ef34af49e84f83210c94c64abacc6536ed432caa4da99bdca1c1c8a7ea
27f07394cbd704a35d8ff552f86889d0e5339e5b6182b46155ad97b20267ef57091bbf64bc0e535d85a68a04047c209645026d7826302770d7ca3595ee6
7873cf3dfd601a1af
}

Meaning of each of those integers:

Encoding explanation: For all of these Integers in DER encoding, the first byte (2 hex values) represents the type, which is x02 or INTEGER, according to the Wikipedia page on X.690 type table. Following that, for some integers, we have a length prefix that tells how many following bytes contain the information about the length of the value. If there is no prefix, it means that the length is encoded in one byte from 0x00 to 0x7F. If the prefix is 81, it means that the length of the value is encoded in one byte (and is bigger than 0x75). If the prefix is 82, it means that the length of the value is stored in the next two bytes. Following the prefix, we have the actual length of the number. Finally, we have the value of the integer.

For example, for the version number, we have 02 01 00 which means that it is a type INTEGER with a length of 1 and a value is 0.
For the second value we have
02 82 01 81 00 D8 B3 51 74 76 FE DF FA B4 2D BD 15 32 BB D7 B3 F0 2C 3D E7 81 E5 EF 9E A3 40 07 20 5D 23 67 31 CB 31 ED B0 C8 02 F9 F7 2C 81 D2 42 25 D2 03 68 2E 66 8D 7D DD F3 8C 8D D5 09 11 3E 4D 75 34 5D F6 DB E0 BC … skipping 288 bytes … 33 4F 8B CA 9E 6C 19 27 81 31 DF 83 38 99 0B 3B 72 2B F3 7E 12 EF 8C 00 84 E5 49 FF, which means that it is an INTEGER, and its length is encoded in the next 2 bytes with value 0x0181 = 385, meaning that the length of the value is 385 bytes.

1. Version Number: According to the RFC 8017, this should be 0x00 (meaning that it conforms with the version of the RFC 8017). The value has length=2+1.

Offset: 4
Value: 0x00

2. Modulus: The decimal used to modulo in RSA. Length = 4 + 385

Offset: 7
Value:
0x00d8b3517476fedffab42dbd1532bbd7b3f02c3de781e5ef9ea34007205d236731cb31edb0c802f9f72c81d24225d203682e668d7dddf38c8dd509113e4
d75345df6dbe0bc204d2ed629efb564f024ae306b80b516d206ab545b81ffb8d26ad8a283257e1aa057148871efab0d573147524e029662f5b01fbc318faf
7c89a06d4bb6a0c4268dca48d5598fa967191057dca6be4c0257ed91bc2263c6d8cd834de9f3658e5034f8d4e25bdd77dc1fada219dd394cdfce10b05a9
cfd8d4f59958075a5a6ed8195402a23c38ab1015809105d096c4d1f9aeb4159b7362eb8f987d924ea9e78b03eac6738d931f1108b74a8e8e17063679090
e202ebcaf4c4bd045abf1389f0f32476c423ebb6a60b4cd17cb6c5d3de3fb61dc06c752b50fa514286f317d7e9ce2a49c1fcb9eeb51c51561a6950079447f
7699702703afe280b334cead4bccb624345cadaf25221b78919ae772522da6bd64df3c5bf66cf9da627682b6a5959a3334f8bca9e6c19278131df8338990
b3b722bf37e12ef8c0084e549ff

3. publicExponent: An exponent **e** used in RSA. Length = 2+3

Offset: 396
Value: 0x010001

4. privateExponent: An exponent d used in RSA. Length = 4+385

Offset: 401
Value:
0x009b473b19fa5ebe3e5637e422b97cf29ee47dbf4a11c1a30788b14ba9fc336f31b609abe99e18cbe73b043796e0a4f17e08a9c1bfac78658208627bf1a
9ed4f3df9b32b01831feb6cd916046434d403e17c56f8ec46493321ba8497aa6754dab0f0cabfe6f27813a586071809f22dc48e8c89bc1b6333f639934603
9878c1f52f68c9206503fb2c5f7cd2bdafbbbd0348804f220617dd8b2d45733bbc84df0ec892797615b12f90d436c1a6ae9c207e648c2a1014db03086b95c
cf62222ee46e2349244b5b8d1bc9128c64aaa10d52b346142f08e063dd1cb5ddcef7c46c93fa880dd1fee622cff389a4ac09df3a68244d9155b632b0c59a5
16e85e098392dace328a64d752f80c2a7d96fc11d6eed807447a9fbc3c492b30210f6460c1daf21861e2e223b56376f169b176c7d13ab7d2bd954fc84d55
bc2dcabded5a2086d20074938da3eeae6e6f9de25b11d28a3723572d97a3941734308176177a01ad43ca3329e6268fcb613508baa1c4693014bb4282c4
9be1b1f3a765fbe909a08da259

5. prime1: The prime factor **p** of n used in RSA. Length = 3+193

Offset: 790
Value:
0x00f65cafec9df2b28c1427c4e281fa0d0ceaefe2b7fa43305fe48c1be5caa428bfabcd0517d8aaec709f4a945ec4aefab2a1678e6796d50a1d41b1ba7aa9
057c97a8bba42fd46329cd8870df95df7618037777fb19ec2aff4db8e688520bfb98ef59e9f21c5bcd71ed827a4e2345b2e00df9acf7d1c8b0b83c28da0c06
7bd35149eecef2b468ec79247b5dc326b49d5e312f5510cbad679abd335954a86c42387d6a40a0e475bec1fa2f924345332bb8f4598392828cde90d38e4
176642d660e6b

6. prime2: The prime factor **q** of n used in RSA. Length = 3+193

Offset: 986
Value:
0x00e12d920a188da0921caab3fdd8f252875ada30c1d4f51ab5ffd25038dd894c4f758209a3f467452a3e68fa996de9b83ef16e3d977ccab75c765a1d7d9
9cbe81b9a2f844edf96189bf2b7609cc7c37c9bfe514f7fc1ed597a4b6dc0ee94a3a142112ce72de30d727ad5563b634d8232b35b9641f5f33f775c2183e8f
ff408b87772d2fc8bd5b964902d9f972d4643fdfe48ca339d6cb72e6c022f6a498653b00a27c81f3ba9dbfd4cbd7600a643993098fb7969020a3f32fb3b90a
87347752fbd

7. exponent1: d mod (p - 1) in RSA. Length = 3+192

Offset: 1182
Value:
0x7fe1bf04413cf1079b92c5359aa894cba6cc5cd0895d052851b794e7a457b03b464fc71c08c138b88147325b3b1765555024981f72da96a75a35b41645
8865b9551e60d2532c742b4c8e5667a22a3e7752d379fa5661d2fa7459a8d1aa1d3b9204d790dcf40ac153e67a35b1a1e1ceb06f61a9e0d4c72c73f9b29
0c62f535cd15669c100a2c08cc1d6ed84debb287999448708ca65d5dd149f2f4a4e84441df96e267a7dccf6df4bd0ca28cbec8002322dbb05b85dedfcd27b
3c9314e70ee6fff

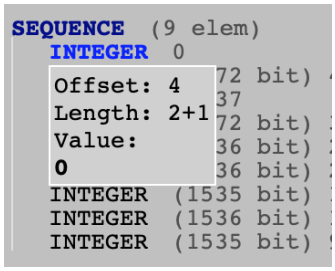
8. exponent2: d mod (q - 1) in RSA. Length = 3+193

Offset: 1377
Value:
0x008caf9e18b4b277ea12030d25c28fd3c4419d24ed559235b81437606bc6edeb4191d74712aab682de1bd4170e193ff71f4c546622e992f5f0ccf90c87d
6b931315da9e66e28ff6c2cb2fea35c10eb30457e256b18151fa7d1e2a295f1f9e829ee4f30d0d7388ec441be5c7c0a54f8cd9885d6612bbb68a1b86c3030
390a4d108380b815958058ae5dee43517eb5f1f11518814cb244a4465ab11ad2bb5133a43f140fd283c26271e5aa51b9a22785e479105510a877a278c5
c2b764a771b3e3b9

9. coefficient: The certificate coefficient q^(-1) mod p. Length = 3+192

Offset: 1573
Value:
0x60e86d61abf7e6cdb4e88a082a81a75f9613395d4cdad945a78e66dd5ee39cd0b17299db9470b93274910e7e27bb8c966a42582c155c98586672af5c
2588c9b49ea8f02bfe018fae3105dd7021967ce61b9c665b64724aae76d17d7e49ef34af49e84f83210c94c64abacc6536ed432caa4da99bdca1c1c8a7ea
27f07394cbd704a35d8ff552f86889d0e5339e5b6182b46155ad97b20267ef57091bbf64bc0e535d85a68a04047c209645026d7826302770d7ca3595ee6
7873cf3dfd601a1af

Note: We are getting the offset from decoding the private key in the Lapo decoder.



Public key

From the RFC 8017 Appendix 1.1, we expect to find 2 integers and a version to be in the file in this order:

1. modulus
2. publicExponent

Decoded the private key file with Lapo ASN1.JS Decoder

To decode the public key, we first needed to get it in the correct format, so we typed in the terminal this command: `ssh-keygen -e -f id_rsa_homework.pub -m PKCS8` (export the openSSH public key

file id_rsa_homework.pub to PEM format that uses PKCS8). Then we copied the output (on page 1) and pasted it onto the decoder. We got the following result:

```
SEQUENCE (2 elem)
 SEQUENCE (2 elem)
  OBJECT IDENTIFIER 1.2.840.113549.1.1.1 rsaEncryption (PKCS #1)
  NULL
 BIT STRING (3184 bit) 001100001000001000000001100010100000001010000010000000011000000100000…
  SEQUENCE (2 elem)
   INTEGER (3072 bit) 491775117665278954444099748349574814605450719657746150480049380822527…
   INTEGER 65537
```

In this decoding, what we care about is the bit string, which contains the sequence with the two integers that we expect.

Meaning of each of those integers:

Encoding explanation: As with the private key, the first byte (2 hex values) represents the type, which is x02 or INTEGER for the last two integers. Following that, for the first integer, we have a length prefix, telling how many following bytes hold information about the length of the value. Then, following the prefix (if any) we have the actual length of the number. Finally, we have the value of the integer.

1. Modulus is the RSA modulus n. Length: 4+385

Offset: 28
Value:
0x00d8b3517476fedffab42dbd1532bbd7b3f02c3de781e5ef9ea34007205d236731cb31edb0c802f9f72c81d24225d203682e668d7dddf38c8dd509113e4
d75345df6dbe0bc204d2ed629efb564f024ae306b80b516d206ab545b81ffb8d26ad8a283257e1aa057148871efab0d573147524e029662f5b01fbc318faf
7c89a06d4bb6a0c4268dca48d5598fa967191057dca6be4c0257ed91bc2263c6d8cd834de9f3658e5034f8d4e25bdd77dc1fada219dd394cdfce10b05a9
cfd8d4f59958075a5a6ed8195402a23c38ab1015809105d096c4d1f9aeb4159b7362eb8f987d924ea9e78b03eac6738d931f1108b74a8e8e17063679090
e202ebcaf4c4bd045abf1389f0f32476c423ebb6a60b4cd17cb6c5d3de3fb61dc06c752b50fa514286f317d7e9ce2a49c1fcb9eeb51c51561a6950079447f
7699702703afe280b334cead4bccb624345cadaf25221b78919ae772522da6bd64df3c5bf66cf9da627682b6a5959a3334f8bca9e6c19278131df8338990
b3b722bf37e12ef8c0084e549ff

2. publicExponent: An exponent **e** used in RSA. Length: 2+3

Offset: 417
Value: 65537

Sanity check

The integers we found work as expected from an RSA key pair as the following relationships hold:

We know from the private key that
e = 0x010001 = 65537

d =

0x009b473b19fa5ebe3e5637e422b97cf29ee47dbf4a11c1a30788b14ba9fc336f31b609abe99e18cbe73b043796e0a4f17e08a9c1bfac78658208627bf1a
9ed4f3df9b32b01831feb6cd916046434d403e17c56f8ec46493321ba8497aa6754dab0f0cabfe6f27813a586071809f22dc48e8c89bc1b6333f639934603
9878c1f52f68c9206503fb2c5f7cd2bdafbbbd0348804f220617dd8b2d45733bbc84df0ec892797615b12f90d436c1a6ae9c207e648c2a1014db03086b95c
cf62222ee46e2349244b5b8d1bc9128c64aaa10d52b346142f08e063dd1cb5ddcef7c46c93fa880dd1fee622cff389a4ac09df3a68244d9155b632b0c59a5
16e85e098392dace328a64d752f80c2a7d96fc11d6eed807447a9fbc3c492b30210f6460c1daf21861e2e223b56376f169b176c7d13ab7d2bd954fc84d55
bc2dcabded5a2086d20074938da3eeae6e6f9de25b11d28a3723572d97a3941734308176177a01ad43ca3329e6268fcb613508baa1c4693014bb4282c4
9be1b1f3a765fbe909a08da259 =

352384932186080610642070407285111965282002094173480888241658894560427280796206075241159103733
501511604965912121778900524483021620355464279662513594902309865828048182208855594616038977974
906839163644594339877949987956917526836040897067702030563262483583899054386596999816467988017
256039420104687554412300233660551741880698578630860909977778563278353597746608647348688468002
806970058555976560470923673260112486788401246539673308763060836763191040187515732593372391545
763480753640427193648052742680872843907883302496425993693123913434564559636591471585601500882
141462030320737099035266705476454257860393339495923809083954770557262210351938711172322056513
867289726717038441658258982033199674514629949317592053585720835121548377096644607918661628403
387479609313471044766064863035254868747559854861956009329464604473282697136585630321882651752
566356664360517229220509401025910498051420822342230318825045807389146756359411673108540 1
n =

491775117665278954444099748349574814605450719657746150480049380822527686836757257151249849478
982316519125465444198884258704963436324525936335302771855639396025910728421919658958526149348
213826755558352233823411093476129212671229579249291709653212950887715174875628874533588776446
134218936466448931102807018875483475812596468297751995426282951844582946179159109075424205777
133374432562829493485720593140009626001393762791903273038120580683812355096616036066490586994292564310595880271064619874205705911157029019959469652751301283555274376792290524119126126
753717830377464395209122340944264391618177649548425976788034333493968592527373639369830392304483976446457701387141114849045540642703157984932287280845756344823501963674803333829928374609003904041101362699461090871284403291353411072071330351021938980731323080897488276240812951669023006692289341396967055752529455078503715523285158767696384679949173496953998151155796479
lambda(n) = lcm(p-1, q-1)

Then:

- By running a python code >>> e * d % math.lcm(p-1, q-1) == 1, we get True. Thus, we conclude that e*d mod lambda(n) = 1.
- Check that p*q == n.
    - p*q =

      491775117665278954444099748349574814605450719657746150480049380822527686836757257151249849478982316519125465444198884258704963436324525936335302771855639396025910728421919658958526149348213826755558352233823411093476129212671229579249291709653212950887715174875628874533588776446134218936466448931102807018875483475812596468297751995426282951844582946179159109075424205777133374432562829493485720593140009626001393762791903273038120580683812355096616036066490586994292564310595880271064619874205705911157029019959469652751301283555274376792290524119126126753717830377464395209122340944264391618177649548425976788034333493968592527373639369830392304483976446457701387141114849045540642703157984932287280845756344823501963674803333829928374609003904041101362699461090871284403291353411072071330351021938980731323080897488276240812951669023006692289341396967055752529455078503715523285158767696384679949173496953998151155796479

      = n
- The modulus from the public key (int 1) is equal to the modulus in the private key (int 2).
- The public exponent from the public key (int 2) is equal to the public exponent of the private key (int 3).

Citations

We used the documents/pages linked in the assignment.
https://crypto.stackexchange.com/questions/21102/what-is-the-ssl-private-key-file-format For length encoding.
https://www.di-mgt.com.au/rsa_alg.html#:~:text=To%20compute%20the%20value%20for,is%20known%20as%20modular%20inversion%20. For what lambda is.