Kaitlyn Peterson

Contents of id_rsa_homework:

-----BEGIN RSA PRIVATE KEY-----
MIIG5AIBAAKCAYEAwTP1czedgssnAAUv5d7SHWLXqpEfdHKl++6mzOyzp1BVK/+j
XjLMUUyzDnQX8TepwNqMijmmYdB18UZcyeDhW1ZXUptiwBfBOOq4L+wozB4P+FEL
rFlRHqmqnxgJY+oe77Q5qgd5Dj8uyq9v7v9wIYCzivw0KR4LHX0YfY+dk8Lt84A6
J5JLIm80xy9pWqHo5dqk+dgHIkc6RZR9IzWzIbO6Q6cWA3WylIFKqpSYeH6ytTF4
xTW+gbORb8LMXU9osgqoA5hxfn05J8zrwWAS9VblbBstsFJ5PNQ84mQFOmt0lfGI
lFRLhhGBqx6z7O8BH540yronMrSfD1unddA557K8iDIHbxMCVjm1/HvM5IkuK+T7
tJTBo8LY6BlguGzF6nAh4uAOatYCEa3HVJq8TNu1tmsFTXcyXv4yyogGPjHr+x+v
RWsL0Kj+uQNeoIOrIosBaYpfxxoaN5meEFejfiiJDpXpqC9jSVbd31veKDFr/ldF
6LAoLy17GTc/QCsfAgMBAAECggGBAIf59r0xnlUlgBlejTh+22i9QUaVvwqpCjNG
4vSFf2e5fUhrMmLA49YwYjFA+fzWjU3jQ1ihfH2JRwkZ2YJoGJO9L9Y+8IB7940E
fb+UuZ33ZowBFnMlpZcsGJ7CPxGoHD/em1sUyVmClH8ofGI8O4II39Ro7pklwPLv
MUZlxqTRY8GmNMfXCbRhikbKEp8sJwtnGAeTyGmSset4t54JVFra8LReq3qh4p7s
ZE2sKadcnJA0EmjSC5Ojey+qbUUa9j0RzP2VaRNOd7EtmOShEPau0Dc7x2ookl2K
uhFZzmHgq3HrEjBnNyyuT6PcT7NyJRVGZ9CEHyMOj+qtZpJK8dYjJGMZ3y4xyNp+
owjSgwtXcKE8iQVk3er1ZqEFfx1YR2SY7S+E5TABMAqZgkp3vmH0Zc6VZfihj4n+
Y1N4DYa5xBLgqjrfAubaRihjdVv4zESNoXaKdJgq+jrOBwcsXc6XCabbFMXwEpaC
1yM2U6nG9tvcMwIFv1tsSQu1qQdvGQKBwQD1UmI2u+r8AsFTno6ZsgMR1PfV/i/h
XXm75niA82+P7mXWvfh5PU7yg4Gdl5pHZjSiXoEvuk+3Ei+2kPRo6Rz3SLzea2hc
EYxPxbciShfMpTtp3tRzqOR2nAKjUlfrFNlb6tsi/EhH8cL9kOl1UXKlhJEannhI
Hy9khkBT0A/M89lhC7X5FukdYbMfIdJt+l6iYiYG6SeqkrRALQZai6Q9vyp0Nw9g
ZIhRrvYYqzIW/EleDxX/9nOHE9Nwo2xE420CgcEAyZzQ2HyatUxHpP9v5YRZeUpP
i8dnndahX2k3I+2nIk32gZS4odv8SGXJpkMedynxrD6irZCf3p4HPdrwDyc9FFnv
BcSWphWhRTNBfBVCFY/uM/p/IoSw1lnzUj84yK9I3gTbyO4y4XKmyiHHd2jlohHN
CeVbXRgSuJa+OJrdA0rsHoRNayiHhlrPPbe2Tcw3Kfjb+9YOQ/h1OZuuTyaFF+bq
9gKiN+oQ/qb4J3+hv7pj9pjs+pCdrZo1yVG+5SU7AoHBAKwUzOXd7Vg3SoswVpED
KFoSzIIkGkv4eNLQOMkjeac5r9/xQ+EvmzhL/NihpkvdKqCWsAHdq9sb+7IRfhA1
LOdc83cFWp/ygzzV0L9Rv94CLWn3L4mt+AwnJUaRFTDGYC200WB6HO2ybXfWdZyg
tDmE3BATBwZ4MZNPBMKF2P+IMXx8bXo7kuvUQ4RcAjK5BDbdqBUC6Zsh6yR1MNbE
y0Gw3cPFd5jDDEhX9TKaq7kp4QEEjgcNRj6DLrb9O44bBQKBwA+uE4wSvNGG2jeZ
6jD2hyLoxaEAZC2haLGL9E7mB86iM57GW0mzWz6iM/mrVK49497ajDplehmNPtDv
uyXijlyL1gwjyvelzQIdx2UIHjihFgyS2dpsaXhyTHtEEX2CLG+f/xv0cp6YFSK5
V9MQpHNjYQf4/48Q9TvH9Ylfefrsk8qXzZGc+FVZsK6DIDmSvWOVSD6g2kEtDKjr
U7YyL725JuL1N3qTyLiVcc1YntLTgzzFjgC6yJic2rwKkIcvywKBwHGzu4okQbHp
0PHPx81di64BQGxVl1tIDt2WsOXyHEbRtKK+d29OkNUK7zdhgMLT59spGjvEbg+u
qcvtrZW04Q1V9Yi6Lc2JKLFLXft1K7Y/0VtH8gn6zYUU5rAgB9tR9eQQKZORIMgs
ca5dM9b+Kf0Ga/IndsP59hmvtjsYk0XEuWxSyPx+NSuaPs47pfW/zP7SpXVyjejW
WKRpn5vRfvkzWO/DMM0caGKovioO22OQJU39IROKQGBWAePCxJIkLQ==
-----END RSA PRIVATE KEY-----

Contents of id_rsa_homework.pub:

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDBM/VzN52CyycABS/l3tIdYteqkR90cqX77qbM7LO
nUFUr/6NeMsxRTLMOdBfxN6nA2oyKOaZh0HXxRlzJ4OFbVldSm2LAF8E46rgv7CjMHg/4UQu
sWVEeqaqfGAlj6h7vtDmqB3kOPy7Kr2/u/3AhgLOK/DQpHgsdfRh9j52Twu3zgDonkksibzTHL2la
oejl2qT52AciRzpFlH0jNbMhs7pDpxYDdbKUgUqqlJh4frK1MXjFNb6Bs5FvwsxdT2iyCqgDmHF+
fTknzOvBYBL1VuVsGy2wUnk81DziZAU6a3SV8YggVEuGEYGrHrPs7wEfnjTKuicytJ8PW6d10
DnnsryIMgdvEwJWObX8e8zkiS4r5Pu0lMGjwtjoGWC4bMXqcCHi4A5q1gIRrcdUmrxM27W2aw
VNdzJe/jLKiAY+Mev7H69FawvQqP65A16gg6siiwFpil/HGho3mZ4QV6N+KIkOlemoL2NJVt3fW9
4oMWv+V0XosCgvLXsZNz9AKx8= kaitlynpeterson@Kaitlyns-MacBook-Air.local

Private Key:
In the private key file (id_rsa_homework), from the PKCS #1: RSA Cryptography Specifications Version 2.2 Documentation, I expect the following items to be found:
Within a RSAPrivateKey structure of the type sequence, I expect to first see the **version**, which will be an integer: likely 0, unless the multi-prime version is used, then the integer will be 1.
Next, I will see the **modulus**. This is an integer that has the value of *n* in RSA.
Then, there will be the **publicExponent**, which is an integer representing the public value *e* in RSA.
Following is the **privateExponent**, which is an integer that represents the private exponent *d* in RSA.
Next is **prime1**, which is the integer prime factor (*p*) of *n*.
Then, there will be **prime2**, which is the integer prime factor (*q*) of *n*.
Next is the integer **exponent1**, calculated with the equation: exponent1 = d mod (p-1).
Following is the integer **exponent2**, calculated with the equation: exponent2 = d mod (q-1).
Then comes the **coefficient**, which is an integer called the CRT coefficient, calculated by taking the inverse of q mod p.
Then we may see **otherPrimeInfos**. This is optional in the private key file (if the version is 0, it will not be included), and is of type OtherPrimeInfos. It is a sequence that contains all additional primes in order. If these exist, each prime r_3 and beyond will have integer values for the prime number, the exponent (calculated d mod (r_i -1)), and the coefficient
(calculated r_1 * . . . r_{i-1} mod r_i).

To decode the private key file, I first removed the headers and footers on the file "-----BEGIN RSA PRIVATE KEY-----" and "-----END RSA PRIVATE KEY-----". Then, I pasted the rest of the contents of the file id_rsa_homework to the Lapo Luchini ASN.1 decoder, selected DER, and clicked decode.

Using this decoder, I can see that the private key consists of a sequence of 9 integers, as expected.

The first integer is the **version**. Its decimal value is 0. Its offset is 4; at index 4 begins the DER encoding. The bytes are: `02 01 00` We first see 02 in blue; in binary this is 00000010. According to the DNR Encoding Wikipedia page, the first two zeros represent the tag class (native to ASN.1), the third zero corresponds to being a primitive data type (rather than composite), and the following digits are the type: an integer. Next, in green is the length octet, denoted as 01. This is in short form (it is only one octet), and thus the 8th bit is 0, and the first seven bits represent the length of the following value. Thus, we know this integer is one byte long. Finally, we see the value in black (decoded from base64): 00.

The second integer is the **modulus.** Its decimal value is:
438450428944728806903142304281093132846444856513246605653763181611240587091769040152802693119791227192625785786310651841979779378669375630232260965496105215245881194713417992615441676080912144435070533680568514376993771335496095469640812558781186502252703718925614674028926107808717405270261749122518017750092957050817927949468026449834218717875800822057612268581369812757368700360253659944640634225184822709789429737236571229220351232159712065636247697401826898651111157151841184588790837960374349368479801707529279427361693912265849542954990755316668635901697270631817734424292515815090964877080389865604946647152735296633862270254607293223520044857805536370485048706700101181885621835552707256745159815026463413597045944708521400940477502729220641659234036503545029959835938910473845750883058237486441709854008100900555720932429999063036314911772084703708216504832800538135513768572294557507184657867542224740776585620150 3
Its offset is 7. Thus at index 7 begins the bytes representing this integer. Of the first few bytes (starting at 7) we see: `02 82 01 81 . . .` 02 in blue represents the type (integer) as explained earlier, and 82 01 81 in green denotes the length of the following value (385 bytes). This is the long form; there are three octets, the first beginning with a 1 to signify that it is in long form, and the rest of the 23 bits encode (in binary) the length. Then, following the colored bytes, are 385 bytes in black that represent the value of the integer.

The third integer is the **publicExponent.** Its decimal value is: 65537
Its offset is 396; at index 396 begin the bytes representing this integer. Of the first few bytes (starting at 396) we see: `02 03 01 00 01` 02 in blue represents the type (integer) as explained earlier, and 03 in green denotes the length of the following value (3 bytes). Then, after the colored bytes, are 3 bytes in black that represent the value of the integer: 01 00 01.

The fourth integer is the **privateExponent.** Its decimal value is:
308581809284459407943717881272646302890615515001808134119349783356248715681337976670400296323456526149501264467302116609111148264966735598890138960182917329954320004061162471724174852514336522301411849276229042001294500859815942193527053103953220736628255169376748658613977092675543438333930805167708982234814496299326745604303106947199342941575313378967718477933315266390491345409718175457322569748945632962891762616995542181482043739923504570356774418155503

3451761406594061630463808004189310375457416697674153479947286334651904897416
1506136393852325355147427922884101019829944289580274624667833159507498008940
3726374732047801652347658505022287529798758679258767233216789796074049831737
9806432908706551145019078507020460273940187311530298024993878723214736612895
3601873129585253936959831399378085277430414053986208967987941860835930177015
8399815505549425113244034535295985291724857897329909513445164200220572438751
364807880473
Its offset is 401; at index 401 begin the bytes representing this integer. Of the first few bytes (starting at 401) we see: `02 82 01 81. . .` 02 in blue represents the type (integer) as explained earlier, and 82 01 81 in green denotes the length of the following value (385 bytes). Then, after the colored bytes, are the 385 bytes in black that represent the value of the integer.

The fifth integer is **prime1.** Its decimal value is:
2309774257602756337845402151191252329109496542206635778240356410973443052601
3087538421562977878281726626549979831207851234941910311542606005569063341671
3303724779910018296970655896106468457404560884620890424027396208602488110981
5631422064934885687196963115536355720822713958911055988986931551198066279465
6535236980132970936332490565766814343023799720821419185846430372733533182236
3803858303223186195248154617882314222094144096216913662251589220956533844263
4945389
Its offset is 790; at index 790 begin the bytes representing this integer. Of the first few bytes (starting at 790) we see: `02 81 C1. . .` 02 in blue represents the type (integer) as explained earlier, and 82 C1 in green denotes the length of the following value (193 bytes). Then, after the colored bytes, are 193 bytes in black that represent the value of the integer.

The sixth integer is **prime2.** Its decimal value is:
1898239308458580805418830228772217229143072811461360961172525391257590592394
0599409413959061219782239883073990695104530487579485675825542156282462379517
5646608782349238605884997987233976571137967178377625320606872333708450554842
3216672972045614462861314643746997666709266695421339273235998159422191822173
0559839017471319729607538571331022199150749703452876775967974839142229843118
9679648865114892375096592127397146447203929964671688035737032021743180486370
0084027
Its offset is 986; at index 986 begin the bytes representing this integer. Of the first few bytes (starting at 986) we see: `02 81 C1. . .` 02 in blue represents the type (integer) as explained earlier, and 82 C1 in green denotes the length of the following value (193 bytes). Then, after the colored bytes, are 193 bytes in black that represent the value of the integer.

The seventh integer is **exponent1.** Its decimal value is:
1620193667642039025391625834145796432877499207192597362733836223337353717306
7849416799329716893395932904300305519331921343996743197459833997314729250346
7160314507182865421518196167049307436338939353752948009261385646362588781145
5122939770619980010164206347320167368691593808155700670914418272733361382628
3711207092385870682425819366142274168227826982710247057273696547979511633437

426838534045612543420896464498936275850115046839476536868284798017292243091134942 77
Its offset is 1182; at index 1182 begin the bytes representing this integer. Of the first few bytes (starting at 1182) we see: `02 81 C1. . .` 02 in blue represents the type (integer) as explained earlier, and 82 C1 in green denotes the length of the following value (193 bytes). Then, after the colored bytes, are 193 bytes in black that represent the value of the integer.

The eighth integer is **exponent2.** Its decimal value is:
147631502131824562693131783207226318216308987595076930880210597360879186557708218547969771785460472827080928529796828954476391410180024701116272808868884793439578197669892457060094440022976805502038267048577007374198530902006199751825413713116207900943295225268887913959020805996098016141761425987025918428770769904101983096280916273363474295545068035612421879409749843764507851103010081368614768212745064631176778833544170625742953433656049592569844658594066379
Its offset is 1378; at index 1378 begin the bytes representing this integer. Of the first few bytes (starting at 1378) we see: `02 81 C0. . .` 02 in blue represents the type (integer) as explained earlier, and 82 C0 in green denotes the length of the following value (192 bytes). Then, after the colored bytes, are 192 bytes in black that represent the value of the integer.

The ninth integer is the **coefficient.** Its decimal value is:
107053725508012211411247928755937039302268169968525078996052009733629171021546960023681208606900988013082565422958233394272119630255151860412283942112461877154710361178522603876371260272911071631200315322445095754810657593377806800133644179372810167733151859031435187916493053384472982648023898776614691703064377321254978411071831571166134748846005704031160401034726645179137677728845102111141939270605809008202239500995612472826421137509619070492591703936480977 3
Its offset is 1573; at index 1573 begin the bytes representing this integer. Of the first few bytes (starting at 1573) we see: `02 81 C0. . .` 02 in blue represents the type (integer) as explained earlier, and 82 C0 in green denotes the length of the following value (192 bytes). Then, after the colored bytes, are 192 bytes in black that represent the value of the integer.

Public Key:
From the PKCS #1: RSA Cryptography Specifications Version 2.2 Documentation and Leonardo Giordani's Public key cryptography: RSA keys blog post, I expect the following items to be found in the id_rsa_homework.pub file:
First, there will be the **string "ssh-rsa"**. Then, there will be a mpint (multiple precision integer) that denotes the value of *n* in RSA, then another mpint with the value of *e* in RSA.

To decode the public key, I first converted the file to PEM/PKCs format using the following command in the terminal:
ssh-keygen -e -f id_rsa_homework.pub -m PKCS8

This produced the contents:
MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEAwTP1czedgssnAAUv5d7S
HWLXqpEfdHKl++6mzOyzp1BVK/+jXjLMUUyzDnQX8TepwNqMijmmYdB18UZcyeDh
W1ZXUptiwBfBOOq4L+wozB4P+FELrFlRHqmqnxgJY+oe77Q5qgd5Dj8uyq9v7v9w
IYCzivw0KR4LHX0YfY+dk8Lt84A6J5JLIm80xy9pWqHo5dqk+dgHIkc6RZR9IzWz
IbO6Q6cWA3WyIIFKqpSYeH6ytTF4xTW+gbORb8LMXU9osgqoA5hxfn05J8zrwWAS
9VblbBstsFJ5PNQ84mQFOmt0lfGIIFRLhhGBqx6z7O8BH540yronMrSfD1unddA5
57K8iDIHbxMCVjm1/HvM5IkuK+T7tJTBo8LY6BlguGzF6nAh4uAOatYCEa3HVJq8
TNu1tmsFTXcyXv4yyogGPjHr+x+vRWsL0Kj+uQNeoIOrIosBaYpfxxoaN5meEFej
fiiJDpXpqC9jSVbd31veKDFr/ldF6LAoLy17GTc/QCsfAgMBAAE=

Then, I was able to use the Lapo Luchini ASN.1 decoder by selecting DER and clicking decode.

As expected, the public key consisted of a bit string and two integers. The first integer denotes *n* in RSA. Its value is:
438450428944728806903142304281093132846444856513246605653763181611240587091769040152802693119791227192625785786310651841979779378669375630232260965496105215245881194713417992615441676080912144435070533680568514376993771335496095469640812558781186502252703718925614674028926107808717405270261749122518017750092957050817927949468026449834218717875800822057612268581369812757368700360253659944640634225184822709789429737236571229220351232159712065636247697401826898651111157151841184588790837960374349368479801707529279427361693912265849542954990755316668635901697270631817734424292515815090964877080389865604946647152735296633862270254607293223520044857805536370485048706700101181885621835552707256745159815026463413597045944708521400940477502729220641659234036503545029959835938910473845750883058237486441709854008100900555720932429999063036314911772084703708216504832800538135513768572294557507184657867542224740776585620150 3
Its offset is 28. Thus, starting at index 28 I can find the bytes representing this integer. At this index I see: `02 82 01 81. . .` The 02 represents the type (integer) as explained earlier. The 82 01 81 is the length of the following integer in bytes (385 bytes). Then, there are 385 bytes that represent the value of the integer above.

The second integer denotes *e* in RSA. Its value is 65537. Its offset is 417. Thus, starting at index 417 I can find the bytes representing this integer. At this index I see: 02 03 01 00 01. The 02 represents the type (integer) as explained earlier. The 03 is again the length of the following integer in bytes (3 bytes). Then, there are 3 bytes that represent the value of the integer above (01 00 01).

Sanity Check:
Firstly, the values for *e* and *n* are equivalent in both the private and public files.

Further, to calculate an RSA key pair, the following calculations are needed (and are defined on the right in terms of the variable names of the private key):

n=pq	modulus = prime1 * prime 2
λ(n) = lcm(p - 1, q - 1)	λ(n) = lcm(prime1 - 1, prime2 - 1)
gcd(e, λ(n)) = 1	gcd(publicExponent, λ(n)) = 1
ed mod λ(n) = 1	publicExponent*privateExponent mod λ(n) = 1

Similarly, according to the documentation of the RSA private key, the exponents are calculated using the following equations:

exponent1 = d mod (p-1)	exponent1 = privateExponent mod (prime1 - 1)
exponent2 = d mod (q-1)	exponent2 = privateExponent mod (prime2 - 1)

Thus, each of these 6 equations should hold with the integers that I found. I wrote these equations as test cases in python in a file named cryptest, which is included in this ssh folder.

Each of these equations holds with the integers I found in the private and public key files; the integers I found have the expected relationships of an RSA key pair.