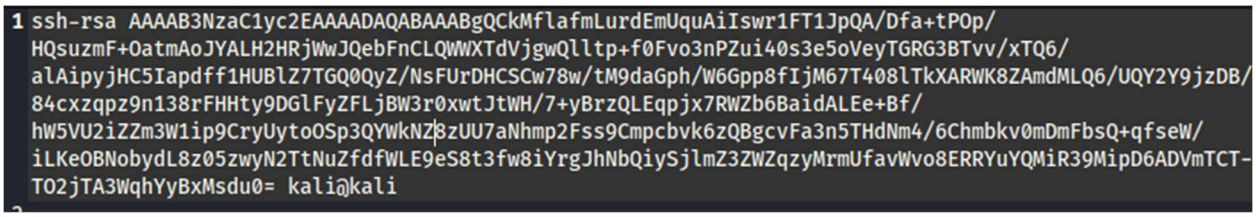
Viktor Chekhovoi

I named my file “homework.” Contents of “homework”:

Contents of “homework.pub”:



Private Key:

I expect the contents of the private key file to contain information about the private key, encoded with DER and then converted to base64. The information should follow the standard ASN.1 syntax defined in RFC 8017:

```
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
}
```

I used Lapo Luchini's ASN.1 decoder to decode the contents of the private key file. After reading the instructions, I copied the contents of the file and pasted them into the text box, then clicked "decode." I got the following output:

```
SEQUENCE  (9 elem)
   INTEGER  0
   INTEGER  (3072 bit) 372620891685503597322444680083102459791792054978738540127139768029752…
   INTEGER  65537
   INTEGER  (3069 bit) 582097546283196641498266419685185922966929682297377843633620236673727…
   INTEGER  (1536 bit) 204379822035784322427601689969659624317294686214920839044059986598593…
   INTEGER  (1536 bit) 182317847218920853659694351363101294146234207175212773623294992593064…
   INTEGER  (1533 bit) 202362126003662735284298543755912126309554178379941304081191579266409…
   INTEGER  (1530 bit) 213094085737359619609267853493653342868937245672235507568921318226783…
   INTEGER  (1534 bit) 311092551821162291661423747065501459379108454818747318706839103406389…
```

It has 9 integers in total:

1. The value of the version is 0, the offset is 4. The DER encoding is 0x020100." judging from the encoding of the other integers and the description of BER, 0x02 indicates that it's an integer, and 0x01 is the number of content octets
2. This is the modulus, also denoted as n. Its value is
   372620891685503597322444680083102459791792054978738540127139768029752899741906640747419757076187365537390374232646403875099562387254053357825088295575920383339057637707832436536963900273955333953322576713424485811067517576223130006963721533755594827130629972490033563942769816889342066149127649869147917863721556098321788374774647629125914445137109416499078648734615378650981126659820396637601432504350115381113087632518052755303265219283538399934204098919369443846198245618560028200292954982483305411055377810276128510442163092333822452852262397252156020048649653228403826253319531221411940715122791835010985601738812352849388066774205870383488251460919238212347556348069140330291349353064889517404034921624318175364185643250876057813169303269211808631061580207296094272397458401903802959173536423908726623552423970863812539750937755325400357853116862284280159388053991302975623049654305450270636569747983104898247219140785 3, the offset is 7. The DER encoding is 0x02820181", followed by 385 bytes. 0x02 indicates that it's an integer, "82" in binary means that it's stored in long form and it's going to be followed by 2 length octets describing its length, and 0x0181 is 385, which is the number of following content octets.

3. This is the public exponent e, its value is 65537 and its offset is 396. Its DER encoding is 0x0203010001. 0x02 means it's an integer, 0x03 means it's stored in short form and has 3 content octets, and the rest are the content octets storing 65537 in hexadecimal.
4. This is the private exponent d, its value is 582097546283196641498266419685185922966929682297377843633620236673727846492460776045910473922518004847918374566097606370946079271359231926608366872164773011371480552184687807691141860476487283368801828034856628428782099416417047623677400714495594830426078346331532359986889449519063135821714280385177286584491400481349233163089986179866504736151139998186912309953917977116551684505430706436938441793114802519467749695854223432380919070910305039676271657954514909461581534219318244035104200977441414402364488379605272930028318167619021688205090013598593753550151732780610625955192024080627913366398576059717624007930172046085739245691277265857268523353146432987431322131835718463493329997602602503904139879977146545089845578515418650551386050444564840751076085275421711796522036777806927716979381023059817770504761390751610731497645400296504625567991422385751481833323422170850083014314725921513563751414821736240809073907145, its offset is 401. Its DER encoding is 0x02820180. 0x02 means it's an integer, 0x82 means it's stored in long form and will be followed by 2 length octets, and 0x0180 means there will be 384 total content octets. It's then followed by those 384 octets which store the value of the exponent.
5. This is the first prime, p. Its value is 204379822035784322427601689969659624317294686214920839044059986598593162480502343275679706691347098194343138703296521602534925466492151393495374598224700423101058191951745124153415925452120393248868916397811595584830656990953082230626822346301661086017094614975538562638739624091367728704530215940832535139863394577821173984510676830748106091059383684624071569625182134265352325479169041211210444547886865633808300997278623646767034824119484065750773207996203446 7, its offset is 789. Its DER encoding is 0x0281c1 followed by 193 more octets. 0x02 means it's an integer, 0x81 means it's stored in long form and has one additional length octet, and 0xc1 is that length octet, which tells us there will be 193 content octets. The following content octets store the value of p.
6. This is the second prime, q. Its value is 182317847218920853659694351363101294146234207175212773623294992593064279581844967136602620858039153587120829895439456458846946412118129989087578232223612723248818729881928190545132362494447039392559906187267142460745366314172077972816169053190054438906408335957034788589017028435304633661159809563962616856674985838928970367171023784264313367374819544706572587586280454333895403227523209721703890359841581821854793771204619795304468138807204011394679699271026295 9, and its offset is 985. Its DER encoding is identical to the previous encoding: 0x0281c1 followed by 193 more octets. 0x02 means it's an integer, 0x81 means it's stored in long form and has one additional length octet, and 0xc1 is that length octet, which tells us there will be 193 content octets. The following content octets store the value of q.
7. This is one of the exponents used to encrypt and decrypt messages more efficiently using the Chinese Remainder Theorem (https://www.di-mgt.com.au/crt_rsa.html), and it's equal to

$d\ mod(p-1)$. Its value is
20236212600366273528429854375591212630955417837994130408119157926640997167035
10544449525636999178052372083902292889709139644981235130644357220327094435328
20468249478550265363173705836437250596120040648565756663220770882587590910568
76839803548399816701480491273269599355521025080929630461627730461267105917612
54813945529392534735770564298732662288998168249386476796018843169310832101450
94895943447925788161053722639908958449249106808211195405969552583631641265875
, its offset is 1181. Its DER encoding is also very similar to the encoding of p and q: 0x0281c0, followed by 192 more octets. 0x02 means it's an integer, 0x81 means its length is stored in long form and it has one additional length octet. 0xc0 means there are 192 content octets. The rest of the encoding are the content octets that store the value of the exponent.

8. This is the other exponent, equal to $d\ mod(q-1)$. Its value is
213094085737359619609267853493653342868937245672235507568921318226783707157320665924039256568439189538328815325551404012201902668237007448679501542461948740419297720508349472752142132948939426850025005934734014564186567277500971553743969810555230938557316913107235070355962347653147610333778497834803797110354516002593330944738093319417220866699897418626477565483758530326020231124834488375765109809174438371516505224137111499158067342610003925611982009615341177, and its offset is 1376. Its DER encoding is identical to that of the first exponent: 0x0281c0, followed by 192 more octets. 0x02 means it's an integer, 0x81 means its length is stored in long form and it has one additional length octet. 0xc0 means there are 192 content octets. The rest of the encoding are the content octets that store the value of the exponent.
9. This is another coefficient for the CRT calculation, equal to $q^{-1}\ mod\ p$. Its value is
311092551821162291661423747065501459379108454818747318706839103406389122096727757763567131856380594836841309359248643504357764095356977551446092013567966789101716904485971987768208452289972674063820703489099535315682226208998894055929246146776381489612026560785828199360445622538045486901652134153804641446036270799160345904699362430283646925590483396048330172862506362088388442902315505053128111951912698528276528800943276908880962925478760918216965661568669668, its offset is 1571. Its DER encoding is similar to the previous encodings: x0281c0, followed by 192 more octets. 0x02 means it's an integer, 0x81 means its length is stored in long form and it has one additional length octet. 0xc0 means there are 192 content octets. The rest of the encoding are the content octets that store the value of the coefficient.

Public key:

After reading the article by Leonardo Giordano, I have to find a way to decode the public key. I converted it to a regular PEM-encoded public key using the following command: ssh-keygen -f key.pub -e -m pem > key.txt. As a result, I had the key in the file key.txt. I expect that key to just contain the public exponent e and modulus n, following the RFC 8017 format:

```
RSAPublicKey ::= SEQUENCE {
    modulus           INTEGER,  -- n
    publicExponent    INTEGER   -- e
}
```

This is the result I get from decoding it:

```
SEQUENCE  (2 elem)
   INTEGER  (3072 bit) 372620891685503597322444680083102459791792054978738540127139768029752…
   INTEGER  65537
```

1. This is the modulus n. It's value is 372620891685503597322444680083102459791792054978738540127139768029752899741906640747419757076187365537390374232646403875099562387254053357825088295575920383339057637707832436536963900273955333953322576713424485811067517576223130006963721533755594827130629972490033563942769816889342066149127649869147917863721556098321788374774647629125914445137109416499078648734615378650981126659820396637601432504350115381113087632518052755303265219283538399934204098919369443846198245618560028200292954982483305411055377810276128510442163092333822452852262397252156020048649653228403826253319531221411940715122791835010985601738812352849388066774205870383488251460919238212347556348069140330291349353064889517404034921624318175364185643250876057813169303269211808631061580200729609427239745840190380295917353642390872662355242397086381253975093775532540035785311686228428015938805399130297562304965430545027063656974798310489824721914078533, it's offset is 4. Its DER encoding is 0x02810181, followed by additional 385 octets. 0x02 means it's an integer, 0x81 means its length is stored in long form with two extra octets for storing the length. 0x0181 means it has 385 content octets. The next 385 octets are those content octets, which store the value of n.
2. This is the public exponent e. Its value is 65537, its offset is 393. Its DER encoding is 0x0203010001. 0x02 means it's an integer, 0x03 means its length is stored in short form and it has 3 content octets. 0x010001 is the 3 content octets, which store 65537.

Sanity Check:

- The public exponents are the same in the public and private key, I tested it with Python.
- $e * d \ mod \ \lambda(n) = e * d \ mod \ lcm(p-1, q-1)! = 1$ when tested with python. I'm not sure why this is failing because I copied all values directly from the ASN.1 decoder in the browser.