// Aadi Akyianu, Lazuli Kleinhans

Private Key

Private Key Contents:

-----BEGIN RSA PRIVATE KEY-----
MIIG4gIBAAKCAYEAl87Gzbf9/8yQRfzm7o0ayYSoTULdQPcx0Pk/fFhJpVIHEQk8
smXX/58dsaBnnH63LP4KW8fsQXDCTA53I5n6utd5xUJE8KRgq+9PX/QSPNvQCgq2
gj/ZP9RNnHQPoGWzFbDZqcrZGYS9l3eWvK4qorvSGD+GmEgGi0BvuO1zQkx1P5wP
lUFXFqaZHfnXbk2UPzQnzIeRnzfh+sGcpTeUmqkCku/9rQggga07Qg0/xFemMIm1
ijGIuPPmMZTL4Ue7iodHFDHqgUBWBAyX3f6GLImwMJaEpffBme9W2WGgMyLpHR3+
ftgEREkEIR8b6X7rrXLQp4/0BSF79qHKuXTFGgZdFrGjvgxZULhR19jqhACt1CkO
/5ye+ZKJFLE0jai9yc/1wIXFfwGHsMVv4p65FdGvbcwZgyRW4HPr0Zu9vnFGxC5K
4c/okfsob2oVgYUQtDqODQFAjt/84urTUPiX3YgGUSfBc/yerr8sWzKxzlYcuYdM
2daVrVHHmEuibq1ZAgMBAAECggGADIkc5gZsQUf/XuQmHqtEu8DKVAzyL8fp02+N
lENHG5F1N20rl7CswDU/srgD7kqRbjPxH4iYBARfav/jyysPJwgdjFIvVq7AM9pw
U8RYEDH3kSi2zvUeR7TMjt+PncdgHSmsI6b2O/L34+7SYfTA3QzE6c8vGlqSPcP9
pcfn1WDyaqUErWbGnf52VOEZbsiVtg1kvvEv73WiIb6EpKj9CrNrnL4YFMZ/xq2L
H+ZvrlMtjEMG3GlAOMOdLdlzZ4Awhgfjfydc0RvWZzXTKrqCaQ4dH2ixDyUWIBUg
xjDjpeNSiOwwI9yppUZA56e5JuP6H4758J7HkKGfKyUGZHc+2RdRjPDLJnvKwuhy
tMFEKYMHplQsVdFxIXfgklWekaq0M6RUxO7cYCs79EthAyQ6kmdLeHq8t6rNo5+V
1u3dySGPI6sBbeCD09T5mjMn3wNy+l3ecH7jYkXdvrR9EHdXkd3vYLODmwjHWNNG
/lEpfsUfN2nTwNi8aiShky3apSDNAoHBAM0SLUNjIlhUmCMA031Ro/ltC2UqxStC
8DSy0yNpRfsc4EUP6+rFXFuoo4jNgfeY4K2P39EuoDfBl+k94NUfXABzplG5w+5H
qEfpJ+UFBhrVcCWA9Vcf3ij0WkwZb/fAmMfc4KoAolfhV1aIeS3R1UJOYwNql7Ss
bFM4pxw1nycpzf8ZN6/R9H+BOzyJgqiWcJg2BmsH2D486HeuZCLJ9PAzQ275ia3b
HnmEvwFwY/r2C8irzigphfCa2BFZJhhcjQKBwQC9gkYaOx9R2ADcJqdXjI5lj5vd
ZL+X7wBia+6jqtoFvYU7N9/74UsWpYx2XynH1r1DyY/e2WCaMbIWD77SVu7fc6DR
16rdYgbGsPpr7bBc3FRWAS26PK6DKAy87FpWh5EeEgwfg3qYADeQqHLYE/2Eo5Pz
Wl3ZCjz1pUF9v01K2rS6tz3wv/yYHup1aR1YzfxZnYiSySyoCztGsbOAhSGHEgbX
fq4akyfKetQ1sZ1J4Rb7eaZHmKMhvjLKj3xZjv0CgcBjWgYR2sp97uYSN0k/Mrl1
ky3tIo6JyhFkBqsiQdOEuFWzP+Z1RI7dXVN1aNMpqKhZjhXKW4OF38YAW1k9B8dj

9KPEfnJ9U7wkssIAQ7HUeEmv8c1rG2ZfgxroOX60kluyoME7u7jrSKIuq0nCOCHQ
0PdJZXAAOXT10MGtehEUQT2q9IW1KcwBphOT3a8ujEwagjoTyqYqaQnbSeL7s8p2
QO/PqRfZ4gnep8B+KvxVTrUg0JO9g3h/vBUoOn+pV9UCgcBHrMT82M2gs9Snb0st
drDIwscNtjQsr61yjCXm6xCoySRh4GErr+spdpWok5eGyLYutEZg39CEoTUc2Pmw
LkcMZnw1wZ8TOicb8QXV2kEw7fhLywhTfu32FyiyL6Z/QajmOacyKBUu9i4N3VgK
V400JHYpvUzvcMrbkZQyji8al5txXyrjKonLsy20GHyMAORntIhaFBJ6wDy8ISul
+TRUFMJXsMcCsBOFpm0qnbHCpop9tGXqgFV6xqgJKrm6WbUCgcBgo6ffDDt/dMGl
QgDOuT8+pbGWmUtIWaxjomzfAC+WLUk3M7BdGuVkaG6Ehe7mEFCyb/SxWKHF71Au
y+pTqzOfd3L4cw8tfnY7etxRJQKMldumvYjKpxiz/auar4fKu9MtHhUDdEFo0at9
rjna4hhAjWKaVNE8rij3jFcGK2tbaQBuXleRQwM63k2LVToSckLlT0tNCEjjGlvd
/LtRUW+hcBASXwtJJXj2Omf98HC8pujoq4u/XCcgP3jyPnKmTfs=
-----END RSA PRIVATE KEY-----

Expected items in id_rsa_homework:

- Version number
- Modulus (n)
- The public exponent (e)
- The private exponent (d)
- Prime 1 (p)
- Prime 2 (q)
- Exponent 1 (d mod (p – 1))
- Exponent 2 (d mod (q – 1))
- The coefficient (q^{-1} mod p)

To decode the private key file, we first removed the header and footer information surrounding the key. We used both the Lapo Luchini ASN.1 Decoder and the Michael Holstrom ASN.1 Decoder to figure out the contents of the private key. We copied the key into the input boxes on those sites and pressed their respective “decode” buttons.

Decoded Integers in id_rsa_homework:

- **Integer 1:**
  - Name: version
  - Value: 0
- **Integer 2:**
  - Name: modulus
  - Value:
    344509002373756900233689480107408228561083483255151485454986567683810241562194225832170792677811813129021771027532701060521886551146526703555790793522802624802018080304603235178587820728673202240259172270284723691348752871343520213277033791055219279016005245963419831599469619946470606095602602876643208254106471753599351869572293491225052814803241026515134724646201121685842079903373515169049383183669912914608960820830822932660976387579829712724632032576800698098904727679072066513336063613842368256829322245005060042378162949257202240860962504962526321190653757086220806929896859905595969359027233468690666744831851086885647365368505328296412062607017960098305017065355856998441466706258907616888524607585541734485434349532693010361124054395411196930582002101287502978396218674840350582260583180104766860554200733494343838097602986141005863932174806636894748573862370643678488235896718258799035011544282050728442450769852 1
- **Integer 3:**
  - Name: publicExponent
  - Value: 65537
- **Integer 4:**
  - Name: privateExponent
  - Value:
    284480002684922853478137394749724046098454848483462174254 39

424411597419393231374668465909139480722030318500531886569341963765804708290216030151682665471826674585674444793575506625610553250792715927225468570629620468691172185075537699345219017396019549489088451447842801141793425779518478689555306432608789983684003121004813771729851856770672604962687111757943544930023044743517034912822080902436782950480663070251060044420800552787569853443756165792260895333951446148565883831789902162256923938918254251855353341418276739531605659746838218754573390037089201891202031824438204690304586557080315977425480239518166680355493529146341935823074361289566005711328204966544270458515279969787607474072511176931981362573549803890374736538940882943004280594180126007349364997092849389194504610966568950681477527095455130652040060082801532745800468153081208644742054368839251100870657363507478392042552283828945636245471890686456608378 9

- **Integer 5:**
  - Name: prime1
  - Value:
    193080151297958266739034536713472089413272428873637088703965741815475587722962656531725290337981137574151356584452637580273752844489961170461073405909608095374524567507382683294909420890113337983728774210006850930072372748993833901171999188834747001878573677326276143011295096820540378661198761481026344773515591974809762303900244580749209979174379365162103183862129812244505408273551227520414495530105525097872891512577205079226584799479572200889019159598685461 89

- **Integer 6:**
  - Name: prime2
  - Value:
    178427974112220370695351495575164035829946906898981676775160481772184565774637342610875179215061626182923836697809492 2

00278733433425778303281381329373177198133127067974316392214
12042596025802546229018634143222142973876701855207423448295
41782682350808120096093033170101014300319202186212527642553
78591881533596871141386347645385086487797268474937909430204
63829926418682076831816099358903149882544143523064580587113
01219018503990840906350584956007364041872858779389

- **Integer 7:**
    - Name: exponent1
    - Value:
      93542394126393837484643568902901449730088543710680244799725
      59421980202611947124994337259400828904434314783592341663023
      65270294118650648812626385356521654429747551694909273780810
      33141930237706857521282785936608692505423000035143814884906
      94156309444384799316855922514255331043824846690615556431874
      88086840874458032549834082901470414702180701052482901602548
      08457940779051310754416298201782962399964535511318397632204
      46707033255573543097308022902336500714656115813 97

- **Integer 8:**
    - Name: exponent2
    - Value:
      67483928076042637417423402365405663306481742852206521846070
      20250678454662032036576736443638255844070061389963270524868
      65176765127229620947470222776625866794349118915098246855030
      19714028364369396429297173277392504675752904296050537103147
      61458008061016047854167571476767908427607620823298277717317
      04099511355915840667339579029580258292724856750886668759095
      22399924644074742272338626243058482405148494057738645194740
      41124221685219185123779231472993047058288073998 61

- **Integer 9:**
    - Name: coefficient

- Value:
  909886164630098849638236030886174500458871390239749157719768455119589084605151595579821805491047713434673238473952313236174342502270594785725521430418153514250955818576436800700802571561005584871699843711430322090774272772238231323405851771362578075574526421295246533879754906881371032378235798436034437762354225114998244872492597498782370036935773251644372535539816349061240416304575811006692750845314275801116357878270066575646656999725872847619688889481317883

Public Key

Public Key Contents:

ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQCXzsbNt/3/zJBF/ObujRrJhKhNQt1A9zHQ+T98W
EmlUgcRCTyyZdf/nx2xoGecfrcs/gpbx+xBcMJMDncjmfq613nFQkTwpGCr709f9BI829AKCra
CP9k/1E2cdA+gZbMVsNmpytkZhL2Xd5a8riqiu9IYP4aYSAaLQG+47XNCTHU/nA+VQVcWp
pkd+dduTZQ/NCfMh5GfN+H6wZylN5SaqQKS7/2tCCCBrTtCDT/EV6YwibWKMYi48+YxlMv
hR7uKh0cUMeqBQFYEDJfd/oYsibAwloSl98GZ71bZYaAzIukdHf5+2ARESQQhHxvpfuutctCnj
/QFlXv2ocq5dMUaBl0WsaO+DFlQuFHX2OqEAK3UKQ7/nJ75kokUsTSNqL3Jz/XAhcV/AYew
xW/inrkV0a9tzBmDJFbgc+vRm72+cUbELkrhz+iR+yhvahWBhRC0Oo4NAUCO3/zi6tNQ+Jfd
iAZRJ8Fz/J6uvyxbMrHOVhy5h0zZ1pWtUceYS6JurVk=

Expected items in id_rsa_homework.pub:

- Modulus (n)
- The public exponent (e)

To decode the public key, we used option 1 from the assignment which was to use ssh-keygen to turn the public key into a .pem file. Then we put it into the Lapo Luchini ASN.1 Decoder to figure out the values.

Decoded Integers in id_rsa_homework.pub

- Integer 1:
    - Name: modulus
    - Value:
      344509002373756900233689480107408228561083483255151485454986567683810241562194225832170792677811813129021771027532701060521886551146526703555790793522802624802018080304603235178587820728673202240259172270284723691348752871343520213277033791055219279016005245963419831599469619946470606095602602876643208254106471753599351869572293491225052814803241026515

13472464620112168584207990337351516904938318366991291460896
08208308229326609763875798297127246320325768006980989047276
79072066513336063613842368256829322245005060042378162949257
20224086096250496252632119065375708622080692989685990559596
93590272334686906667448318510868856473653685053282964120626
07017960098305017065355856998441466706258907616888524607585
54173448543434953269301036112405439541119693058200210128750
29783962186748403505822605831801047668605542007334943438380
97602986141005863932174806636894748573862370643678488235896
71825879903501154428205072844245076985 21

- Integer 2:
    - Name: publicExponent
    - Value: 65537

Sanity Check

We wrote the following Python script:

```
from math import lcm, gcd

n =
344509002373756900233689480107408228561083483255151485454986567683810241562194225832170792677811813129021771027532701060521886551146526703555790793522802624802018080304603235178587820728673202240259172270284723691348752871343520213277033791055219279016005245963419831599469619946470606095602602876643208254106471753599351869572293491225052814803241026515134724646201121685842079903373515169049383183669912914608960820830822932660976387579829712724632032576800698098904727679072066513336063613842368256829322245005060042378162949257202240860962504962526321190653757086220806929896859905595969359027233468690666744831851086885647365368505328296412062607017960098305017065355856998441466706258907616888524607585541734485434349532693010361124054395411196930582002101287502978396218674840350582260583180104766860554200733494343838097602986141005863932174806636894748573862370643678488235896718258799035011544282050728442450769852 1
e = 65537
d =
284480002684922853478137394749724046098454848483462174254394244115974193932313746684659091394807220303185005318865693419637658047082902160301516826654718266745856744447935755066256105532507927159272254685706296204686911721850755376993452190173960195494890884514478428011417934257795184786895553064326087899836840031210048137717298518567706726049626871117579435449300230447435170349128220809024367829504806630702510600444208005527875698534437561657922608953339514461485658838317899021622569239389182542518553353341418276739531605659746838218754573390037089201891202031824438204690304586557080315977425480239518166680355493529146341935823074361289566005711328204966544270458515279969787607474072511176931981362573549803890374736538940882943004280594180126007349364997092849389194504610966568950681477527095455130652040060082801532745800468153081208644742054368839251100870657363507478392042552283828945636245471890686456608378 9
p =
193080151297958266739034536713472089413272428873637088703965741815475587722962656531725290337981137574151356584452637580273752844489961170461073405909608095374524567507382683294909420890113337983728774210006850930072372748993833901171999188347470018785736773262761430112950968205403786611987614810263447735155919748097623039002445807492099791743793651621031838621298122445054082735512275204144955301055250978728915125772050792265847994795722008890191595986854618 9
```

```
q =
178427974112220370695351495575164035829946906898981676775160481772184565774637342610875179215061626182923836697809492200278733433425778303281381329373177198133127067974316392214120425960258025462290186341432221429738767018552074234482954178268235080812009609303317010101430031920218621252764255378591881533596871141386347645385086487797268474937909430204638299264186820768318160993589031498825441435230645805871130121901850399084090635058495600736404187285877938 9

lambda_n = lcm((p-1), (q-1))

print("Does p * q == n?\t\t", str(p * q == n))
print("Does (e * d) % lambda_n == 1?\t", str((e * d) % lambda_n == 1))
print("Does gcd(e, lambda_n) == 1?\t", str(gcd(e, lambda_n) == 1))
```

And got the following output:

```
Does p * q == n?                 True
Does (e * d) % lambda_n == 1? True
Does gcd(e, lambda_n) == 1?    True
```