

Author: Tianyi Lu (Sky)

Content

Private Key Content

-----BEGIN RSA PRIVATE KEY-----

MIIG4wIBAABKAYEA5MHmlo3BmeSHs6GsEEyJIthudc1f+7g7lnCXTzWXIx8Ml7cD
PpPldlJDAd9q0ZA4wF6uJr+upe0ar0jckrjsjE1sL9QmnD6j5K8wCNCtyqZpLeaA
3Vtkcyjz05vHigUho9xmxdwio5FiMeqaDPPW51oDqgPEkB0Vugsyx+zdGCAGquCE
Kzj/DIXDPJdDIjfnCpJJ5S6RxhdFWzTEBaLDqcKkKpavH893UwguF89p5IS0NctvL
+KM0VB82J+6k2tbPSRNoQZxYv0wZjPDY5WeAsmjC7SxgfYkg5NCNhSSImo+bKqi
f0eL6zS7WMMT0WLZDziCmNE6ZFPWu6Myx1BpL1Rj8kUzky0RUg4/M4mt6hAi+2Pf
vBUFG3qBBH2F16GCLzXNoPzILVkg+ayonxbhL+AmGdiLWgJ0H4FnHJNTI0lVwAVc
pQ007CcW24tjN5DcKchV22DhjCzKXArxCar4zlfNFL9pX6h/Fmo3A201IhL11PR9
LX04H6o7+9UckdydAgMBAAECggGASZnRdRM6/Sa40qWD10KrJNj6AVEI1taZ5Xdo
V4XokNZGmPG7S4E0TRL+kbWuiqqad9p+sdzudlhpQ4hv3408HpAyhNj/azoERWnQ
TKXHpPIR2UDWqWHkWeENQkJW2yN3hV8+ed6wDmySaXkQA6bYuCsQDr2Tmr+SZv9/
YS/mmwKC4qvGGpoGBaWDZJV0dk35/ijvCDVqiSGnifGAgIybgZAB6vfwJ5hyAdTv
Tlw2q0Dle9UYNXFdpSukAwkXZYnDH/fvZCVFQ+bqMki0Sck73HUGwRsmQ0zbV5vU
k5cUUPTpysvN+0oHWlNKNQiohRY34aJmsgiGxq6KCgcdy0Esj0ao4/SuhBLl9dH1
BI4kqsBiEiXX06D3Cyjaxq5Sncvw/cm9r9KhQh9beoJ1nFWAts9Gjb0/t6zyysOB
nVsad8Feuku53qy47ppph/+ZYWl4Xd+hRXGyfmqSZ7XF9Mdo+LJmgCNBnwjDuMxz
hh8tCcFT0Ytvq9ZCKPw3VC+j3JLJAoHBAPkp4Tj16iWdWU/mw4MVC5AstqMUZZwJ
MKpfpUwrVs0BbWhfvvQ67nRzBwgPj4lJvvnvKNgQfnFTmEtD73XyzkqpKWbD6uJk+
i4oBiiQFm+nAMppWeEtmflvyh5ftM788ZLhtwPb1x3BEzgzUxfZdy0Vw69Rmubn9
DuLN0AEuSxiq4Gh0woCS4C6k0TfcJ7nrFk6ws20b0jnavSfMgQKD6lzo9CBcrynF
UxAAKK0fzk1CvBSS9CuZjmxpxdER/ru0WwKBwQDrCLA9EN1y0kGSTpMiRfT0kxUg
VKSblG3CEDdGAwtN6nJGGa9fiWPW7YzgzTANWiRpZWh5mueFfml7dMW59JQQR3a
3nVVaz0qLzN8/LXXjh0sDADV0qW9ijXB07vVKKUBthyfVvLj1Rf8ZRZFndzSUF+g
7wV5HQmHk3X+qTEXUfc6WUM1tb/556uFBwq4pCxIgT7NsU5zKRQlKbSLLckRy9Aa
heplVhTeX3zMqMoYT8Hc8fJl2CXBoL0koafyomcCgcEAmKcn7ZX8ln8IsQNY1X6X
TwWV/+QHPMoEnt1qkHgc/gzZ0xCWFVvlrEKtFCs0bZBYu87vuScEyJ8M+CzXTgdW
3C4g0oJdhsXjB/JZiChHlbbDSfugR9HLqNm2aUZSZd/CdZwIlUWPoyEEelVxzVE4
PBoq4j7tmx2HeT1PKuoJ6FsIkfa3E77oyqr/45FmZWm6iQuJXIVF7xwst617f0cF
N1ch+p8j3Ixyj2MJgvXtLzhDsvc10WCK+bD0GVtFDfPxAoHAM695cjRfPltpSA9X
c9YGhDJEfXQGDuZCZmm/2d8L7nWrNLnU/h7bi6cTbkiQNSSJll3fhfbx+5XBjQVKh
bqYULeo/bf41t08X0XkHkZiVNI3gBGuvUzT0ddwK7LDtoXaKstGlgbTRPRLwb2Sb
fTu179oU7YMipPupuNLiW/PZA9PWllJC4XDSgtZbdamSxf/0bQ1fDkTLh6+k6nc9
aM3sxakYXnzMivRfJNbpruf/aVFAxKHA0H1L8ES003VpRYZJAoHARvtPb0BqMFPo
0vVmNICKbVC03QS2TCZASvBJu90GD0o8bUCucJ2wvHbI+Vuj8EuwkIKD3ZeNf7ti
KnaZoxH/e0/yzAZFF3taf+mUX8xV1Zkit2zTk3sxVzH2MerVKsrqx85XnN5AHCGF
avE5h0B6nno7Bw8iqrnQxcNCAcjRkdSD8RaUEZCcw6lbtfx1oLaMz/SYsXDAU/gT

```
Mw7h3mIjM3yy6u86tbcQ8PKMHXdp0vdtfxRbYuhZYVz0HJ0DMvw6
-----END RSA PRIVATE KEY-----
```

Public Key Content

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQgQDkweaWjcGZ5IezoawQTiki2G51zV/7uDuWcJdPNZcjHwyXtwM+k+V2UkM
B32rRkDjAXq4mv66l7RqvSNySu0yMTWwv1CacPqPkrzAI0K3Kpmkt5oDdW2RzKPPTm8eKBSGj3GbF3CKjKWix6p
oM89bnWg0qA8SQE5W6CzLH7N0YIAaq4IQR0P8MjEM8l0MiN81ykkLLPHGF0VbNMQFos0pwqQqlq8fz3dTCC4Xz2
nkhI41y28v4ozRUHzYn7qTa1s9JE2hBnFi87BmM8NjlZ4CyaMLtLGB9iSDk0I2HFJIiaj5sqqJ/R4vrNLtYwxPR
YtkP0IKY0TpkU9a7ozLHUGkvVGPYRT0TI5FSDj8zia3qECL7Y9+8FQVDeAEEfYWxoYIvNc2g/MgtWSD5rKifFuE
v4CYZ2ItaAk4fgWcck1MjSVXABVylDQ7sJxbbi2M3kNwpyFXbY0GMLMpcCvEJqvj0V+d+X2lfqH8WajcDY7UiEv
XU9H0tc7gfgqv71RyR3J0= lutianyi@ichis-MacBook-Air.local
```

Private Key

Expectation

According to RFC 8017, the PEM file of a RSA private key should contain the following:

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

Decoding

Using the decoder provided by [Lapo Luchini's ASN.1 decoder](#) , we just have to paste our private key content in the website, select the DER format, and hit decode to get the following content:

ASN.1 JavaScript decoder

clear OS Theme

```
SEQUENCE (9 elem)
  INTEGER 0
  INTEGER (3072 bit) 519136916517588390204475394572021244456546544762820664480755929371862...
  INTEGER 65537
  INTEGER (3071 bit) 167028106591541706148459163067361032098062780457891519771140264686132...
  INTEGER (1536 bit) 234594571967677615933426963033690325915545766924925154938317618347784...
  INTEGER (1536 bit) 221291103269480144028393602180777322359756351680411711376227761842611...
  INTEGER (1536 bit) 143727074074891918076185352087046004030715376559341972032337900909413...
  INTEGER (1534 bit) 486633105012397246706014708428729232318966162689487401674503639726524...
  INTEGER (1535 bit) 668312606942666789283233668424802831943557535026608622629042898399625...
```

```
bqYULeo/bf41t08X0XkHzzIvNI3gBGuvUzTOddwK7LdtoXaKstGlgbTRPRLwb2Sb
fTu179oU7YMipPupuNLiW/PZA9PW11JC4XDSgtZbdamSxf/0bQ1fDkTLh6+k6nc9
aM3sxakYXnzMivRfJNbpruf/avFAxKHAOH1L8ES003VpRYZJaoHARvtPb0BqMFPo
OvVmNICkbVCO3QS2TCZASvBjU9OGD0o8bUCucJ2wvHBI+VuJ8EuwIKD3ZneF7ti
KNaZoxH/e0/yzA2FF3taf+mUX8xV1Zkit2zTk3sxVzH2MerVKsrqx85XnN5AHCgf
avE5h0B6nno7Bw8iqrnQxeNCacjrKdSD8RaUEZCcw6lbtfx1oLaMz/SYsXDAU/gT
Mw7h3mIjM3yy6u86tbcQ8PKMHXdp0vdtfxRbYuhZYYVz0HJ0DMVw6
-----END RSA PRIVATE KEY-----
```

☒ with hex dump ☐ trim big chunks ☐ with definitions

Drag or load file:

Load examples:

Definitions:

Instructions

This page contains a JavaScript generic ASN.1 parser that can decode any valid ASN.1 DER or BER structure whether Base64 encoded or Hex-encoded.

This tool can be used online at the address <http://lapo.it/asn1js/> or offline, unpacking [the ZIP file](#) in a directory and opening the index.html file.

On the left of the page will be printed a tree representing the hierarchical structure, on the right side an hex dump will be shown. Hovering on the tree highlights ancestry (the hovered node and all its ancestors get colored) and the position of the hovered node (different colors).

Clicking a node in the tree will hide its sub-nodes (collapsed nodes can be noticed because they will become *italic*).

WARNING: starting from 2023-02-26 this website is using some ES6 features, which can break it for older browsers (though it works on all modern browsers). You can access [last version before ES6 on github](#).

```
30 82 06 E3 02 01 00 02 82 01 81 00 E4 C1 E6 96 8D C1 99 E4 87 B3 A1 AC 10 4C 89 22 D8
6E 75 CD 5F FB B8 3B 96 70 97 4F 35 97 23 1F 0C 97 B7 03 3E 93 E5 76 52 43 01 DF 6A D1
90 38 C0 P0 5E AE 26 BF AE A5 ED 1A AF 48 DC 92 B8 EC 8C 4D 6C 2F D4 26 9C 3E A3 E4 AF 30
08 D0 AD CA A6 69 2D E6 80 DD 5B 64 73 28 F3 D3 9B C7 8A 05 21 A3 DC 66 C5 DC 22 A3 91
62 31 EA 9A 0C F3 D6 E7 5A 03 AA 03 C4 90 13 95 BA 0B 32 C7 EC DD 18 20 06 AA E0 84 2B
38 FF 0C 8C 43 3C 97 43 22 37 CD 72 92 49 4B A4 71 85 D1 56 CD 31 01 68 B0 EA 70 A9 0A
A5 AB C7 F3 DD D4 C2 0B 85 F3 DA 79 21 23 8D 72 DB CB F8 A3 34 54 1F 36 27 EE A4 DA D6
CF 49 13 68 41 9C 58 BC EC 19 8C F0 D8 E5 67 80 B2 68 C2 ED 2C 60 7D 89 20 E4 D0 8D 87
14 92 22 6A 3E 6C AA A2 7F 47 8B EB 34 BB 58 C3 13 D1 62 D9 0F 38 82 98 BB A3 32 C7
50 69 2F 54 63 F2 45 33 93 23 91 52 0E 3F 33 89 AD EA 10 22 FB 63 DF BC 15 05 43 78
01 04 7D 85 97 A1 82 2F 35 CD A0 FC C8 2D 59 20 F9 AC A8 9F 16 E1 2F E0 26 19 D8 D8
8B 5A 02 4E 1F 81 67 1C 93 53 23 49 55 C0 05 5C A5 0D 0E EC 27 16 DB 8B 63 37 90 DC
29 C8 55 DB 60 E1 8C 2C CA 5C 0A F1 09 AA F8 CE 57 E7 7E 5F 69 5F A8 7F 16 6A 37 03 63
B5 22 12 F5 D4 F4 7D 2D 73 B8 1F AA 3B FB D5 1C 91 DC 9D 02 03 01 00 01 02 82 01 80 49
99 D1 75 13 3A FD 26 B8 3A A5 83 D4 E2 AB 24 D8 FA 01 51 08 D6 D6 99 E5 77 68 57 85 E8
90 D6 46 98 F1 BB 4B 81 0E 4D 19 7E 91 B5 AE 8A AA 9A 77 DA 7E B1 DC EE 76 58 69 43 88
6F DF 8D 3C 1E 90 32 84 D8 FF 6B 3A 04 45 69 D0 4C A5 C7 A4 F2 11 D9 40 D6 A9 61 E4 59
E1 0D 42 42 56 DB 23 77 85 5F 3E 79 DE B0 0E 6C 92 69 79 10 03 A6 D8 B8 2B 10 0E BD 93
9A BF 92 66 FF 7F 61 2F E6 9B 02 82 E2 AB C6 1A 9A 06 05 A5 83 64 95 4E 76 4D F9 FE 28
EF 08 35 6A 89 21 A7 89 F1 80 80 8C 9B 81 90 01 EA F7 F0 27 98 72 01 D4 EF 4E 5C 36 A8
E0 E5 7B D5 18 35 71 5D 3E CB A4 03 09 17 65 89 C3 1F F7 EF 64 25 45 43 E6 EA 32 48 8E
49 C9 3B DC 75 06 C1 1B 26 40 EC DB 57 9B D4 93 97 14 50 F4 E9 CA CB CF FB 4A 07 5A 53 4A 35
08 A8 85 16 37 E1 A2 66 B2 08 86 C6 AE 8A 0A 07 1D CB 41 2C 8F 46 A8 E3 F4 AE 84 12 E5 F5 D1 F5
04 8E 24 AA C0 62 12 25 D7 3B A0 F7 0B 28 DA C6 AE 52 9D CB F0 FD C9 BD AF D2 A1 42 1F 5B 7A 82
```

The decoded bytes is as follows:

```
30 82 06 E3 02 01 00 02 82 01 81 00 E4 C1 E6 96 8D C1 99 E4 87 B3 A1 AC 10 4C 89 22 D8
6E 75 CD 5F FB B8 3B 96 70 97 4F 35 97 23 1F 0C 97 B7 03 3E 93 E5 76 52 43 01 DF 6A D1
90 38 C0 P0 5E AE 26 BF AE A5 ED 1A AF 48 DC 92 B8 EC 8C 4D 6C 2F D4 26 9C 3E A3 E4 AF 30
08 D0 AD CA A6 69 2D E6 80 DD 5B 64 73 28 F3 D3 9B C7 8A 05 21 A3 DC 66 C5 DC 22 A3 91
62 31 EA 9A 0C F3 D6 E7 5A 03 AA 03 C4 90 13 95 BA 0B 32 C7 EC DD 18 20 06 AA E0 84 2B
38 FF 0C 8C 43 3C 97 43 22 37 CD 72 92 49 4B A4 71 85 D1 56 CD 31 01 68 B0 EA 70 A9 0A
A5 AB C7 F3 DD D4 C2 0B 85 F3 DA 79 21 23 8D 72 DB CB F8 A3 34 54 1F 36 27 EE A4 DA D6
CF 49 13 68 41 9C 58 BC EC 19 8C F0 D8 E5 67 80 B2 68 C2 ED 2C 60 7D 89 20 E4 D0 8D 87
14 92 22 6A 3E 6C AA A2 7F 47 8B EB 34 BB 58 C3 13 D1 62 D9 0F 38 82 98 D1 3A 64 53 D6
BB A3 32 C7 50 69 2F 54 63 F2 45 33 93 23 91 52 0E 3F 33 89 AD EA 10 22 FB 63 DF BC 15
05 43 78 01 04 7D 85 97 A1 82 2F 35 CD A0 FC C8 2D 59 20 F9 AC A8 9F 16 E1 2F E0 26 19
D8 8B 5A 02 4E 1F 81 67 1C 93 53 23 49 55 C0 05 5C A5 0D 0E EC 27 16 DB 8B 63 37 90 DC
29 C8 55 DB 60 E1 8C 2C CA 5C 0A F1 09 AA F8 CE 57 E7 7E 5F 69 5F A8 7F 16 6A 37 03 63
B5 22 12 F5 D4 F4 7D 2D 73 B8 1F AA 3B FB D5 1C 91 DC 9D 02 03 01 00 01 02 82 01 80 49
99 D1 75 13 3A FD 26 B8 3A A5 83 D4 E2 AB 24 D8 FA 01 51 08 D6 D6 99 E5 77 68 57 85 E8
90 D6 46 98 F1 BB 4B 81 0E 4D 19 7E 91 B5 AE 8A AA 9A 77 DA 7E B1 DC EE 76 58 69 43 88
6F DF 8D 3C 1E 90 32 84 D8 FF 6B 3A 04 45 69 D0 4C A5 C7 A4 F2 11 D9 40 D6 A9 61 E4 59
E1 0D 42 42 56 DB 23 77 85 5F 3E 79 DE B0 0E 6C 92 69 79 10 03 A6 D8 B8 2B 10 0E BD 93
9A BF 92 66 FF 7F 61 2F E6 9B 02 82 E2 AB C6 1A 9A 06 05 A5 83 64 95 4E 76 4D F9 FE 28
EF 08 35 6A 89 21 A7 89 F1 80 80 8C 9B 81 90 01 EA F7 F0 27 98 72 01 D4 EF 4E 5C 36 A8
E0 E5 7B D5 18 35 71 5D 3E CB A4 03 09 17 65 89 C3 1F F7 EF 64 25 45 43 E6 EA 32 48 8E
```

```

49 C9 3B DC 75 06 C1 1B 26 40 EC DB 57 9B D4 93 97 14 50 F4 E9 CA CB CD FB 4A 07 5A 53
4A 35 08 A8 85 16 37 E1 A2 66 B2 08 86 C6 AE 8A 0A 07 1D CB 41 2C 8F 46 A8 E3 F4 AE 84
12 E5 F5 D1 F5 04 8E 24 AA C0 62 12 25 D7 3B A0 F7 0B 28 DA C6 AE 52 9D CB F0 FD C9 BD
AF D2 A1 42 1F 5B 7A 82 75 9C 55 80 B6 CF 46 8D B3 BF B7 AC F2 CA CA 01 9D 5B 1A 77 C1
5E BA 4B B9 DE AC B8 EE 9A 69 87 FF 99 61 69 78 5D DF A1 45 71 B2 7E 6A 92 67 B5 C5 F4
C7 68 F8 B2 66 80 23 41 9F 08 C3 B8 CC 73 86 1F 2D 09 C1 53 D1 8B 6F AB D6 42 2A 95 B7
54 2F A3 DC 92 C9 02 81 C1 00 F9 29 E1 38 F5 EA 25 9D 59 4F E6 C3 83 15 0B 90 2C B6 A3
14 65 9C 09 30 AA 5F A5 4C 2B 56 CD 01 6D 68 5F BE F4 3A EE 74 73 07 08 0F 8F 89 49 BE
7B CA 36 04 1F 9C 54 E6 12 D0 FB DD 7C B3 92 AA 4A 58 17 7A B8 99 3E 8B 8A 01 8A 24 05
9B E9 C0 32 9A 56 78 4B 66 7E 5B F2 87 97 ED 33 BF 3C 64 B8 6D C0 F6 F5 C7 70 44 CE 09
14 C5 F6 5D C8 E5 70 EB D4 66 B9 B9 FD 0E E2 CD D0 01 2E 4B 18 AA E0 68 74 C2 80 92 E0
2E A4 D1 37 DC 27 B9 EB 16 4E B0 B3 63 9B 3A 39 DA BD 27 CC 81 02 83 EA 5C E8 F4 20 72
72 BC 85 53 10 00 28 A3 9F CE 4D 42 BC 14 92 F4 2B 99 8E 6C 69 C5 D1 11 FE BB 8E 5B 02
81 C1 00 EB 08 B0 3D 10 DD 72 3A 41 92 4E 93 22 45 F4 F4 93 15 20 54 A4 9B 94 6D C2 10
37 46 03 0B 4D EA 72 46 19 AF 5F 89 63 D6 ED 8C E0 CD 30 0D 5A 24 69 65 68 79 9A E7 85
7E 69 E5 ED D3 16 E7 D2 50 42 BD DA DE 75 55 6B 33 AA 2F 33 7C FC B5 D7 8E 1D 2C 0C 00
D5 3A A5 BD 8A 35 C1 D3 BB D5 28 A5 01 B6 1C 9F 56 F2 E3 95 17 FC 65 16 45 9D DC D2 50
5F A0 EF 05 79 1D 09 87 93 75 FE A9 31 17 51 F7 3A 59 43 35 B5 BF F9 E7 AB 85 07 0A B8
A4 2C 48 81 3E CD B1 4E 73 29 14 25 29 B4 8B 2D C9 11 CB D0 1A 85 EA 65 56 14 C4 C7 7C
CC A8 CA 18 4F C1 DC F1 F2 65 D8 25 C1 A0 BD 24 A1 A7 F2 A2 67 02 81 C1
00 98 A7 27 ED 95 FC 96 7F 08 B1 03 58 D5 7E 97 4F 05 95 FF E4 07 3C CA 04 9E DD 6A 90
78 1C FE 0C D9 D3 10 96 15 5B E5 AC 42 AD 14 2B 34 6D 90 58 BB CE EF B9 27 04 C8 9F 0C
F8 2C D7 4E 07 56 DC 2E 20 3A 82 5D 86 C5 E3 07 F2 59 88 28 47 95 B6 C3 49 FB A0 47 D1
CB A8 D9 B6 69 46 52 65 DF C2 75 9C 08 95 45 8F A3 21 04 7A 5B D7 CD 51 38 3C 1A 2A E2
3E ED 9B 1D 87 79 3D 4F 2A EA 09 E8 5B 08 91 F6 B7 13 BE E8 CA AA FF E3 91 66 65 69 BA
89 0B 89 5C 85 45 EF 1C 2C B7 AD 7B 7C E7 05 37 57 21 FA 9F 23 DC 8C 72 8F 63 09 82 F5
ED 97 38 43 B2 F7 35 D1 60 8A F9 B0 F4 19 5B 45 0D F3 F1 02 81 C0 33 AF 79 72 34 5F 3E
5B 69 48 0F 57 73 D6 06 84 32 44 17 14 06 0E E0 99 32 6F F6 77 C2 FB 9D 6A CD 2E 75 3F
87 B6 E2 E9 C4 DB 92 24 0D 49 22 65 97 77 E1 7D BC 7E E5 70 63 41 59 07 6E A6 14 2D EA
3F 6D FE 35 B7 4F 17 D1 79 07 CD 98 95 34 8D E0 04 6B AF 53 34 CE 75 DC 0A EE 50 ED A1
76 8A B2 D1 A5 81 B4 D1 3D 12 F0 6F 64 9B 7D 3B B5 EF DA 14 ED 83 22 A4 FB A9 B8 D2 C8
C3 F3 D9 03 D3 D6 96 52 42 E1 70 D2 82 D6 5B 75 A9 92 C5 FF F4 6D 0D 5F 0E 44 CB 87 AF
A4 EA 77 3D 68 CD EC C5 A9 18 5E 7C CC 8A F4 5F 24 D6 E9 AE E7 FF 69 51 40 C4 A1 C0 38
7D 4B F0 44 B4 D3 75 69 45 86 49 02 81 C0 46 FB 4F 6F 40 6A 30 53 E8 3A F5 66 34 80 A4
6D 50 8E DD 04 B6 4C 26 40 4A F0 49 BB D3 86 0F 4A 3C 6D 40 AE 70 9D B0 BC 76 C8 F9 5B
A3 F0 4B B0 90 82 83 DD 97 8D 7F BB 62 2A 76 99 A3 11 FF 7B 4F F2 CC 06 45 17 7B 5A 7F
E9 94 5F CC 55 D5 99 22 B7 6C D3 93 7B 31 57 31 F6 31 EA D5 2A CA EA C7 CE 57 9C DE 40
1C 21 85 6A F1 39 87 40 7A 9E 7A 3B 07 0F 22 AA B9 D0 C5 C3 42 01 C8 EB 29 D4 83 F1 16
94 11 90 9C C3 A9 5B B5 FC 75 A0 B6 8C CF F4 98 B1 70 C0 53 F8 13 33 0E E1 DE 62 23 33
7C B2 EA EF 3A B5 B7 10 F0 F9 0C 1D 77 69 D2 F7 6D 7F 14 5B 62 E8 59 61 5C F4 1C 9D 03
32 FC 3A

```

If we want to decode manually, we can first use `base64 --decode` to decode everything in between `-----BEGIN RSA PRIVATE KEY-----` and `-----END RSA PRIVATE KEY-----`. Then, we can use `hexdump -C` to get the same bytes in hexadecimal representation:

```

cs338/ssh on ʝ main [?] >
base64 -D < base64_en.txt | hexdump -C
00000000 30 82 06 e3 02 01 00 02 82 01 81 00 e4 c1 e6 96 |0.....|
00000010 8d c1 99 e4 87 b3 a1 ac 10 4c 89 22 d8 6e 75 cd |.....L."nu.|
00000020 5f fb b8 3b 96 70 97 4f 35 97 23 1f 0c 97 b7 03 |_...;.p.05.#....|
00000030 3e 93 e5 76 52 43 01 df 6a d1 90 38 c0 5e ae 26 |>..vRC..j..8.^.&|
00000040 bf ae a5 ed 1a af 48 dc 92 b8 ec 8c 4d 6c 2f d4 |.....H.....Ml/.|
00000050 26 9c 3e a3 e4 af 30 08 d0 ad ca a6 69 2d e6 80 |&.>...0.....i-..|
00000060 dd 5b 64 73 28 f3 d3 9b c7 8a 05 21 a3 dc 66 c5 |.[ds(.....!..f.|
00000070 dc 22 a3 91 62 31 ea 9a 0c f3 d6 e7 5a 03 aa 03 |."..b1.....Z...|
00000080 c4 90 13 95 ba 0b 32 c7 ec dd 18 20 06 aa e0 84 |.....2....|
00000090 2b 38 ff 0c 8c 43 3c 97 43 22 37 cd 72 92 49 4b |+8...C<.C"7.r.IK|
000000a0 a4 71 85 d1 56 cd 31 01 68 b0 ea 70 a9 0a a5 ab |.q..V.1.h..p....|
000000b0 c7 f3 dd d4 c2 0b 85 f3 da 79 21 23 8d 72 db cb |.....y!#.r..|
000000c0 f8 a3 34 54 1f 36 27 ee a4 da d6 cf 49 13 68 41 |..4T.6'.....I.hA|
000000d0 9c 58 bc ec 19 8c f0 d8 e5 67 80 b2 68 c2 ed 2c |.X.....g..h.,|
000000e0 60 7d 89 20 e4 d0 8d 87 14 92 22 6a 3e 6c aa a2 |`}. ...."j>l..|
000000f0 7f 47 8b eb 34 bb 58 c3 13 d1 62 d9 0f 38 82 98 |.G..4.X...b..8..|
00000100 d1 3a 64 53 d6 bb a3 32 c7 50 69 2f 54 63 f2 45 |.:dS...2.Pi/Tc.E|
00000110 33 93 23 91 52 0e 3f 33 89 ad ea 10 22 fb 63 df |3.#.R.?3....".c.|
00000120 bc 15 05 43 78 01 04 7d 85 97 a1 82 2f 35 cd a0 |...Cx...}..../5..|
00000130 fc c8 2d 59 20 f9 ac a8 9f 16 e1 2f e0 26 19 d8 |..-Y ...../.&..|
00000140 8b 5a 02 4e 1f 81 67 1c 93 53 23 49 55 c0 05 5c |.Z.N..g..S#IU..\\|
00000150 a5 0d 0e ec 27 16 db 8b 63 37 90 dc 29 c8 55 db |....'....c7..).U.|
00000160 60 e1 8c 2c ca 5c 0a f1 09 aa f8 ce 57 e7 7e 5f |`...,\.....W.~_|
00000170 69 5f a8 7f 16 6a 37 03 63 b5 22 12 f5 d4 f4 7d |i_...j7.c."....}|
00000180 2d 73 b8 1f aa 3b fb d5 1c 91 dc 9d 02 03 01 00 |-s...;.....|
00000190 01 02 82 01 80 49 99 d1 75 13 3a fd 26 b8 3a a5 |.....I..u.:.&.:|
000001a0 83 d4 e2 ab 24 d8 fa 01 51 08 d6 d6 99 e5 77 68 |....$.Q.....wh|
000001b0 57 85 e8 90 d6 46 98 f1 bb 4b 81 0e 4d 19 7e 91 |W....F...K..M.~|
000001c0 b5 ae 8a aa 9a 77 da 7e b1 dc ee 76 58 69 43 88 |....w.~...vXiC.|
000001d0 6f df 8d 3c 1e 90 32 84 d8 ff 6b 3a 04 45 69 d0 |o..<..2...k:.Ei.|
000001e0 4c a5 c7 a4 f2 11 d9 40 d6 a9 61 e4 59 e1 0d 42 |L.....@..a.Y..B|
000001f0 42 56 db 23 77 85 5f 3e 79 de b0 0e 6c 92 69 79 |BV.#w._>y...l.iy|
00000200 10 03 a6 d8 b8 2b 10 0e bd 93 9a bf 92 66 ff 7f |.....+.....f..|
00000210 61 2f e6 9b 02 82 e2 ab c6 1a 9a 06 05 a5 83 64 |a/.....d|
00000220 95 4e 76 4d f9 fe 28 ef 08 35 6a 89 21 a7 89 f1 |.NmM..(..5j.!...|
00000230 80 80 8c 9b 81 90 01 ea f7 f0 27 98 72 01 d4 ef |.....'.r...|

```

As a result, we get the following ASN.1 structure:

```

SEQUENCE (9 elem)
  INTEGER 0
  INTEGER (3072 bit)
519136916517588390204475394572021244456546544762820664480755929371862...
  INTEGER 65537
  INTEGER (3071 bit)
167028106591541706148459163067361032098062780457891519771140264686132...
  INTEGER (1536 bit)
234594571967677615933426963033690325915545766924925154938317618347784...
  INTEGER (1536 bit)
221291103269480144028393602180777322359756351680411711376227761842611...

```



```
INTEGER (1536 bit)
143727074074891918076185352087046004030715376559341972032337900909413...
INTEGER (1534 bit)
486633105012397246706014708428729232318966162689487401674503639726524...
INTEGER (1535 bit)
668312606942666789283233668424802831943557535026608622629042898399625...
```

Interpreting Integers

- The first integer `0` is the version number of this RSA private key format. The value `0` means that the version my `sshkeygen` uses is the same as RFC 8017.
 - Offset: 6
 - DER Encoding: `00`
- The second represents the modulus n in the RSA algorithm.:
 - Offset: 11
 - DER Encoding:
"00 E4 C1 E6 96 8D C1 99 E4 87 B3 A1 AC 10 4C 89 22 D8 6E 75 CD 5F FB B8 3B 96 70 97 4F
35 97 23 1F 0C 97 B7 03 3E 93 E5 76 52 43 01 DF 6A D1 90 38 C0 5E AE 26 BF AE A5 ED 1A AF
48 DC 92 B8 EC 8C 4D 6C 2F D4 26 9C 3E A3 E4 AF 30 08 D0 AD CAA6 69 2D E6 80 DD 5B 64
73 28 F3 D3 9B C7 8A 05 21 A3 DC 66 C5 DC 22 A3 91 62 31 EA 9A 0C F3 D6 E7 5A 03 AA 03 C4
90 13 95 BA 0B 32 C7 EC DD 18 20 06 AA E0 84 2B 38 FF 0C 8C 43 3C 97 43 22 37 CD 72 92 49
4B A4 71 85 D1 56 CD 31 01 68 B0 EA 70 A9 0A A5 AB C7 F3 DD D4 C2 0B 85 F3 DA 79 21 23
8D 72 DB CB F8 A3 34 54 1F 36 27 EE A4 DA D6 CF 49 13 68 41 9C 58 BC EC 19 8C F0 D8 E5
67 80 B2 68 C2 ED 2C 60 7D 89 20 E4 D0 8D 87 14 92 22 6A 3E 6C AA A2 7F 47 8B EB 34 BB 58
C3 13 D1 62 D9 0F 38 82 98 D1 3A 64 53 D6 BB A3 32 C7 50 69 2F 54 63 F2 45 33 93 23 91 52
0E 3F 33 89 AD EA 10 22 FB 63 DF BC 15 05 43 78 01 04 7D 85 97 A1 82 2F 35 CD A0 FC C8 2D
59 20 F9 AC A8 9F 16 E1 2F E0 26 19 D8 8B 5A 02 4E 1F 81 67 1C 93 53 23 49 55 C0 05 5C A5
0D 0E EC 27 16 DB 8B 63 37 90 DC 29 C8 55 DB 60 E1 8C 2C CA 5C 0A F1 09 AA F8 CE 57 E7
7E 5F 69 5F A8 7F 16 6A 37 03 63 B5 22 12 F5 D4 F4 7D 2D 73 B8 1F AA 3B FB D5 1C 91 DC
9D"
 - Hexadecimal Value:
"0x00e4c1e6968dc199e487b3a1ac104c8922d86e75cd5ffbb83b9670974f3597231f0c97b7033e93e
576524301df6ad19038c05eae26bfaea5ed1aaf48dc92b8ec8c4d6c2fd4269c3ea3e4af3008d0adcaa
6692de680dd5b647328f3d39bc78a0521a3dc66c5dc22a3916231ea9a0cf3d6e75a03aa03c490139
5ba0b32c7ecdd182006aae0842b38ff0c8c433c97432237cd7292494ba47185d156cd310168b0ea7
0a90aa5abc7f3ddd4c20b85f3da7921238d72dbcbf8a334541f3627eea4dad6cf491368419c58bcec1
98cf0d8e56780b268c2ed2c607d8920e4d08d871492226a3e6caaa27f478beb34bb58c313d162d90f
388298d13a6453d6bba332c750692f5463f24533932391520e3f3389adea1022fb63dfbc150543780
1047d8597a1822f35cda0fcc82d5920f9aca89f16e12fe02619d88b5a024e1f81671c9353234955c00
55ca50d0eec2716db8b633790dc29c855db60e18c2cca5c0af109aaf8ce57e77e5f695fa87f166a370
363b52212f5d4f47d2d73b81faa3bfbfd51c91dc9d
- The third integer is the public exponent e :
 - Offset: 399

- DER Encoding:
"01 00 01"
- Hexadecimal Value:
"0x010001"
- Next, we have the private exponent d as the forth integer
 - Offset: 405
 - DER Encoding:
"49 99 D1 75 13 3A FD 26 B8 3A A5 83 D4 E2 AB 24 D8 FA 01 51 08 D6 D6 99 E5 77 68 57 85 E8 90 D6 46 98 F1 BB 4B 81 0E 4D 19 7E 91 B5 AE 8A AA 9A 77 DA 7E B1 DC EE 76 58 69 43 88 6F DF 8D 3C 1E 90 32 84 D8 FF 6B 3A 04 45 69 D0 4C A5 C7 A4 F2 11 D9 40 D6 A9 61 E4 59 E1 0D 42 42 56 DB 23 77 85 5F 3E 79 DE B0 0E 6C 92 69 79 10 03 A6 D8 B8 2B 10 0E BD 93 9A BF 92 66 FF 7F 61 2F E6 9B 02 82 E2 AB C6 1A 9A 06 05 A5 83 64 95 4E 76 4D F9 FE 28 EF 08 35 6A 89 21 A7 89 F1 80 80 8C 9B 81 90 01 EA F7 F0 27 98 72 01 D4 EF 4E 5C 36 A8 E0 E5 7B D5 18 35 71 5D 3E CB A4 03 09 17 65 89 C3 1F F7 EF 64 25 45 43 E6 EA 32 48 8E 49 C9 3B DC 75 06 C1 1B 26 40 EC DB 57 9B D4 93 97 14 50 F4 E9 CA CB CD FB 4A 07 5A 53 4A 35 08 A8 85 16 37 E1 A2 66 B2 08 86 C6 AE 8A 0A 07 1D CB 41 2C 8F 46 A8 E3 F4 AE 84 12 E5 F5 D1 F5 04 8E 24 AA C0 62 12 25 D7 3B A0 F7 0B 28 DA C6 AE 52 9D CB F0 FD C9 BD AF D2 A1 42 1F 5B 7A 82 75 9C 55 80 B6 CF 46 8D B3 BF B7 AC F2 CA CA 01 9D 5B 1A 77 C1 5E BA 4B B9 DE AC B8 EE 9A 69 87 FF 99 61 69 78 5D DF A1 45 71 B2 7E 6A 92 67 B5 C5 F4 C7 68 F8 B2 66 80 23 41 9F 08 C3 B8 CC 73 86 1F 2D 09 C1 53 D1 8B 6F AB D6 42 2A 95 B7 54 2F A3 DC 92 C9"
 - Hexadecimal Value:
"0x4999d175133afd26b83aa583d4e2ab24d8fa015108d6d699e577685785e890d64698f1bb4b810e4d197e91b5ae8aaa9a77da7eb1dcee76586943886fdf8d3c1e903284d8ff6b3a044569d04ca5c7a4f211d940d6a961e459e10d424256db2377855f3e79deb00e6c9269791003a6d8b82b100ebd939abf9266ff7f612fe69b0282e2abc61a9a0605a58364954e764df9fe28ef08356a8921a789f180808c9b819001eaf7f027987201d4ef4e5c36a8e0e57bd51835715d3ecba40309176589c31ff7ef64254543e6ea32488e49c93bdc7506c11b2640ecdb579bd493971450f4e9cacbcd4a075a534a3508a8851637e1a266b20886c6ae8a0a071dcb412c8f46a8e3f4ae8412e5f5d1f5048e24aac0621225d73ba0f70b28dac6ae529dcbf0fdc9bdaafd2a1421f5b7a82759c5580b6cf468db3bfb7acf2caca019d5b1a77c15eba4bb9deacb8ee9a6987ff996169785ddfa14571b27e6a9267b5c5f4c768f8b2668023419f08c3b8cc73861f2d09c153d18b6fabd6422a95b7542fa3dc92c9"
- The next two are large prime numbers p and q that produce n :
- p
 - Offset: 792
 - DER Encoding:
"00 F9 29 E1 38 F5 EA 25 9D 59 4F E6 C3 83 15 0B 90 2C B6 A3 14 65 9C 09 30 AA 5F A5 4C 2B 56 CD 01 6D 68 5F BE F4 3A EE 74 73 07 08 0F 8F 89 49 BE 7B CA 36 04 1F 9C 54 E6 12 D0 FB DD 7C B3 92 AA 4A 58 17 7A B8 99 3E 8B 8A 01 8A 24 05 9B E9 C0 32 9A 56 78 4B 66 7E 5B F2 87 97 ED 33 BF 3C 64 B8 6D C0 F6 F5 C7 70 44 CE 09 14 C5 F6 5D C8 E5 70 EB D4 66 B9 B9 FD 0E E2 CD D0 01 2E 4B 18 AA E0 68 74 C2 80 92 E0 2E A4 D1 37 DC 27 B9 EB 16 4E B0 B3 63 9B 3A 39 DA BD 27 CC 81 02 83 EA 5C E8 F4 20 72 72 BC 85 53 10 00 28 A3 9F CE 4D 42 BC 14 92 F4 2B 99 8E 6C 69 C5 D1 11 FE BB 8E 5B"
 - Hexadecimal Value:
"0x00f929e138f5ea259d594fe6c383150b902cb6a314659c0930aa5fa54c2b56cd016d685fbef43aee"

747307080f8f8949be7bca36041f9c54e612d0fbdd7cb392aa4a58177ab8993e8b8a018a24059be9c
0329a56784b667e5bf28797ed33bf3c64b86dc0f6f5c77044ce0914c5f65dc8e570ebd466b9b9fd0ee
2cdd0012e4b18aae06874c28092e02ea4d137dc27b9eb164eb0b3639b3a39dabd27cc810283ea5c
e8f4207272bc8553100028a39fce4d42bc1492f42b998e6c69c5d111febb8e5b"

- q

- Offset: 988

- DER Encoding:

"00 EB 08 B0 3D 10 DD 72 3A 41 92 4E 93 22 45 F4 F4 93 15 20 54 A4 9B 94 6D C2 10 37 46 03
0B 4D EA 72 46 19 AF 5F 89 63 D6 ED 8C E0 CD 30 0D 5A 24 69 65 68 79 9A E7 85 7E 69 E5 ED
D3 16 E7 D2 50 42 BD DA DE 75 55 6B 33 AA 2F 33 7C FC B5 D7 8E 1D 2C 0C 00 D5 3A A5 BD
8A 35 C1 D3 BB D5 28 A5 01 B6 1C 9F 56 F2 E3 95 17 FC 65 16 45 9D DC D2 50 5F A0 EF 05 79
1D 09 87 93 75 FE A9 31 17 51 F7 3A 59 43 35 B5 BF F9 E7 AB 85 07 0A B8 A4 2C 48 81 3E CD
B1 4E 73 29 14 25 29 B4 8B 2D C9 11 CB D0 1A 85 EA 65 56 14 C4 C7 7C CC A8 CA 18 4F C1
DC F1 F2 65 D8 25 C1 A0 BD 24 A1 A7 F2 A2 67"

- Hexadecimal Value:

"0x00eb08b03d10dd723a41924e932245f4f493152054a49b946dc2103746030b4dea724619af5f89
63d6ed8ce0cd300d5a24696568799ae7857e69e5edd316e7d25042bddade75556b33aa2f337cfc
b5d78e1d2c0c00d53aa5bd8a35c1d3bbd528a501b61c9f56f2e39517fc6516459ddcd2505fa0ef05791d
09879375fea9311751f73a594335b5bff9e7ab85070ab8a42c48813ecdb14e7329142529b48b2dc91
1cbd01a85ea655614c4c77ccca8ca184fc1dcf1f265d825c1a0bd24a1a7f2a267"

- The rest three integers are $d \bmod (p - 1)$, $d \bmod (q - 1)$, and $q^{-1} \bmod p$ respectively. These integers are for speeding up the decryption process using the Chinese Remainder Theorem.

- $d \bmod (p - 1)$

- Offset: 1104

- DER Encoding:

"00 98 A7 27 ED 95 FC 96 7F 08 B1 03 58 D5 7E 97 4F 05 95 FF E4 07 3C CA 04 9E DD 6A 90 78
1C FE 0C D9 D3 10 96 15 5B E5 AC 42 AD 14 2B 34 6D 90 58 BB CE EF B9 27 04 C8 9F 0C F8
2C D7 4E 07 56 DC 2E 20 3A 82 5D 86 C5 E3 07 F2 59 88 28 47 95 B6 C3 49 FB A0 47 D1 CB A8
D9 B6 69 46 52 65 DF C2 75 9C 08 95 45 8F A3 21 04 7A 5B D7 CD 51 38 3C 1A 2A E2 3E ED 9B
1D 87 79 3D 4F 2A EA 09 E8 5B 08 91 F6 B7 13 BE E8 CA AA FF E3 91 66 65 69 BA 89 0B 89 5C
85 45 EF 1C 2C B7 AD 7B 7C E7 05 37 57 21 FA 9F 23 DC 8C 72 8F 63 09 82 F5 ED 97 38 43 B2
F7 35 D1 60 8A F9 B0 F4 19 5B 45 0D F3 F1"

- Hexadecimal Value:

"0x0098a727ed95fc967f08b10358d57e974f0595ffe4073cca049edd6a90781cfe0cd9d31096155be5
ac42ad142b346d9058bbceefb92704c89f0cf82cd74e0756dc2e203a825d86c5e307f25988284795b
6c349fba047d1cba8d9b669465265dfc2759c0895458fa321047a5bd7cd51383c1a2ae23eed9b1d87
793d4f2aea09e85b0891f6b713bee8caaafe391666569ba890b895c8545ef1c2cb7ad7b7ce7053757
21fa9f23dc8c728f630982f5ed973843b2f735d1608af9b0f4195b450df3f1"

- $d \bmod (q - 1)$

- Offset: 1380

- DER Encoding:

"33 AF 79 72 34 5F 3E 5B 69 48 0F 57 73 D6 06 84 32 44 17 14 06 0E E0 99 32 6F F6 77 C2 FB
9D 6A CD 2E 75 3F 87 B6 E2 E9 C4 DB 92 24 0D 49 22 65 97 77 E1 7D BC 7E E5 70 63 41 59 07
6E A6 14 2D EA 3F 6D FE 35 B7 4F 17 D1 79 07 CD 98 95 34 8D E0 04 6B AF 53 34 CE 75 DC

0A EE 50 ED A1 76 8A B2 D1 A5 81 B4 D1 3D 12 F0 6F 64 9B 7D 3B B5 EF DA 14 ED 83 22 A4
FB A9 B8 D2 C8 C3 F3 D9 03 D3 D6 96 52 42 E1 70 D2 82 D6 5B 75 A9 92 C5 FF F4 6D 0D 5F
0E 44 CB 87 AF A4 EA 77 3D 68 CD EC C5 A9 18 5E 7C CC 8A F4 5F 24 D6 E9 AE E7 FF 69 51
40 C4 A1 C0 38 7D 4B F0 44 B4 D3 75 69 45 86 49"

- Hexadecimal Value:

"0x33af7972345f3e5b69480f5773d6068432441714060ee099326ff677c2fb9d6acd2e753f87b6e2e9
c4db92240d4922659777e17dbc7ee570634159076ea6142dea3f6dfe35b74f17d17907cd9895348de
0046baf5334ce75dc0aee50eda1768ab2d1a581b4d13d12f06f649b7d3bb5efda14ed8322a4fba9b8
d2c8c3f3d903d3d6965242e170d282d65b75a992c5fff46d0d5f0e44cb87afa4ea773d68cdecc5a9185
e7ccc8af45f24d6e9aee7ff695140c4a1c0387d4bf044b4d37569458649"

- $q^{-1} \bmod p$

- Offset: 1575

- DER Encoding:

"46 FB 4F 6F 40 6A 30 53 E8 3A F5 66 34 80 A4 6D 50 8E DD 04 B6 4C 26 40 4A F0 49 BB D3 86
0F 4A 3C 6D 40 AE 70 9D B0 BC 76 C8 F9 5B A3 F0 4B B0 90 82 83 DD 97 8D 7F BB 62 2A 76
99 A3 11 FF 7B 4F F2 CC 06 45 17 7B 5A 7F E9 94 5F CC 55 D5 99 22 B7 6C D3 93 7B 31 57 31
F6 31 EA D5 2A CA EA C7 CE 57 9C DE 40 1C 21 85 6A F1 39 87 40 7A 9E 7A 3B 07 0F 22 AA B9
D0 C5 C3 42 01 C8 EB 29 D4 83 F1 16 94 11 90 9C C3 A9 5B B5 FC 75 A0 B6 8C CF F4 98 B1 70
C0 53 F8 13 33 0E E1 DE 62 23 33 7C B2 EA EF 3A B5 B7 10 F0 F9 0C 1D 77 69 D2 F7 6D 7F 14
5B 62 E8 59 61 5C F4 1C 9D 03 32 FC 3A"

- Hexadecimal Value:

"0x46fb4f6f406a3053e83af5663480a46d508edd04b64c26404af049bbd3860f4a3c6d40ae709db0b
c76c8f95ba3f04bb0908283dd978d7fbb622a7699a311ff7b4ff2cc0645177b5a7fe9945fcc55d59922b
76cd3937b315731f631ead52acaeac7ce579cde401c21856af13987407a9e7a3b070f22aab9d0c5c3
4201c8eb29d483f1169411909cc3a95bb5fc75a0b68ccff498b170c053f813330ee1de6223337cb2ea
ef3ab5b710f0f90c1d7769d2f76d7f145b62e859615cf41c9d0332fc3a"

Public Key

The public key file should contain the public exponent e and the modulus n . From RFC 4253 section 6.6, we know that the file indeed contains the following:

```
string    "ssh-rsa"  
mpint     e  
mpint     n
```

To decode the file, we take the base64 string in the middle and use `base64` and `hexdump` command line tools to decode the string into hexadecimal bytes.

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

Then, we will get this result:

"00 00 00 07 73 73 68 2d 72 73 61 00 00 00 03 01 00 01 00 00 01 81 00 e4 c1 e6 96 8d c1 99 e4 87 b3 a1

ac 10 4c 89 22 d8 6e 75 cd 5f fb b8 3b 96 70 97 4f 35 97 23 1f 0c 97 b7 03 3e 93 e5 76 52 43 01 df 6a d1 90 38 c0 5e ae 26 bf ae a5 ed 1a af 48 dc 92 b8 ec 8c 4d 6c 2f d4 26 9c 3e a3 e4 af 30 08 d0 ad ca a6 69 2d e6 80 dd 5b 64 73 28 f3 d3 9b c7 8a 05 21 a3 dc 66 c5 dc 22 a3 91 62 31 ea 9a 0c f3 d6 e7 5a 03 aa 03 c4 90 13 95 ba 0b 32 c7 ec dd 18 20 06 aa e0 84 2b 38 ff 0c 8c 43 3c 97 43 22 37 cd 72 92 49 4b a4 71 85 d1 56 cd 31 01 68 b0 ea 70 a9 0a a5 ab c7 f3 dd d4 c2 0b 85 f3 da 79 21 23 8d 72 db cb f8 a3 34 54 1f 36 27 ee a4 da d6 cf 49 13 68 41 9c 58 bc ec 19 8c f0 d8 e5 67 80 b2 68 c2 ed 2c 60 7d 89 20 e4 d0 8d 87 14 92 22 6a 3e 6c aa a2 7f 47 8b eb 34 bb 58 c3 13 d1 62 d9 0f 38 82 98 d1 3a 64 53 d6 bb a3 32 c7 50 69 2f 54 63 f2 45 33 93 23 91 52 0e 3f 33 89 ad ea 10 22 fb 63 df bc 15 05 43 78 01 04 7d 85 97 a1 82 2f 35 cd a0 fc c8 2d 59 20 f9 ac a8 9f 16 e1 2f e0 26 19 d8 8b 5a 02 4e 1f 81 67 1c 93 53 23 49 55 c0 05 5c a5 0d 0e ec 27 16 db 8b 63 37 90 dc 29 c8 55 db 60 e1 8c 2c ca 5c 0a f1 09 aa f8 ce 57 e7 7e 5f 69 5f a8 7f 16 6a 37 03 63 b5 22 12 f5 d4 f4 7d 2d 73 b8 1f aa 3b fb d5 1c 91 dc 9d".

The format of this file is a sequence of three length-value pairs. The length contains four bytes indicating the number of following bytes that represent the "value", followed by the actual value. For example, the first pair starts with "00 00 00 07". This means the following 7 bytes "73 73 68 2d 72 73 61" is it's corresponding value. This is also the ASCII value of the string "ssh-rsa". Using the same method, we have:

$e = 0x10001$

$n =$

0x00e4c1e6968dc199e487b3a1ac104c8922d86e75cd5ffbb83b9670974f3597231f0c97b7033e93e576524301df6ad19038c05eae26bfaea5ed1aaf48dc92b8ec8c4d6c2fd4269c3ea3e4af3008d0adcaa6692de680dd5b647328f3d39bc78a0521a3dc66c5dc22a3916231ea9a0cf3d6e75a03aa03c4901395ba0b32c7ecdd182006aae0842b38ff0c8c433c97432237cd7292494ba47185d156cd310168b0ea70a90aa5abc7f3ddd4c20b85f3da7921238d72dbcbf8a334541f3627eea4dad6cf491368419c58bcec198cf0d8e56780b268c2ed2c607d8920e4d08d871492226a3e6caa27f478beb34bb58c313d162d90f388298d13a6453d6bba332c750692f5463f245339323919520e3f3389adea1022fb63dfbc1505437801047d8597a1822f35cda0fcc82d5920f9aca89f16e12fe02619d88b5a024e1f81671c93234955c0055ca50d0eec2716db8b633790dc29c855db60e18c2cca5c0af109aaf8ce57e77e5f695fa87f166a370363b52212f5d4f47d2d73b81faa3bfbdd51c91dc9d

Sanity Check

We first confirm that the e and n in private and public key files are indeed identical. Then, we proceed by calculating n from p and q . The numbers are abbreviated for clearer presentation.

$$\begin{aligned} p \cdot q &= 0x00f929e138f5ea259 \dots \times 0x00eb08b03d10dd72 \dots \\ &= 0xe4c1e6968dc199e487b3a \dots = n \end{aligned}$$

This matches our value of n above. Next we will calculate $\phi(n)$.

$$\phi(n) = (p - 1)(q - 1) =$$

0xe4c1e6968dc199e487b3a1ac104c8922d86e75cd5ffbb83b9670974f3597231f0c97b7033e93e576524301df6ad19038c05eae26bfaea5ed1aaf48dc92b8ec8c4d6c2fd4269c3ea3e4af3008d0adcaa6692de680dd5b647328f3d39bc78a0521a3dc66c5dc22a3916231ea9a0cf3d6e75a03aa03c4901395ba0b32c7ecdd182006aae0842b38ff0c8c433c97432237cd7292494ba47185d156cd310168b0ea70a90aa5abc7f3ddd4c20b85f3da7921238d72dbcbf8a334541f3627eea4dad6cd64e0d6cb959125147eaabb82400c802da8f729c35645eb827860b0f4e630077e5ebe31342b83399fd4276fe6543abef62a08e41b16b689ef5a04b98233b50434c74dffec05a479d21dd8192875c28c1d33501669b73e129fb43aabc96033c787511f00ab00dd97f3af432cc631b3b781edaa482cc26e835b520211cbbd6f34c7e1d3756eec46ec39a9a1ea830c50199929663aa1458df28f90f14bb7bf9455fa32343670c3dcd1741a291931b1cbc8077e36f969c0203f19fb9062175e3abdc

Using Python, we can check that $e \cdot d \equiv 1 \pmod{\phi(n)}$:

```
>>> hex(e)
'0x10001'
>>> hex(d)
'0x49999d175133afd26b83aa583d4e2ab24d8fa015108d6d699e577685785e890d64698f1bb4b810e4d197e91b5ae8aaa9a77da7eb1dcee76586943886fdf8d3c1e903284d8ff6b3a044569d04ca5c7a4f211d940d6a961e459e10d424256db2377855f3e79deb00e6c9269791003a6d8b82b100ebd939abf9266ff7f612fe69b0282e2abc61a9a0605a58364954e764df9fe28ef08356a8921a789f180808c9b819001eaf7f027987201d4ef4e5c36a8e0e57bd51835715d3ecba40309176589c31ff7ef64254543e6ea32488e49c93bdc7506c11b2640ecdb579bd493971450f4e9cacbcdfb4a075a534a3508a8851637e1a266b20886c6ae8a0a071dcb412c8f46a8e3f4ae8412e5f5d1f5048e24aac0621225d73ba0f70b28dac6ae529dcbf0fdc9bdafd2a1421f5b7a82759c5580b6cf468db3bf7acf2caca019d5b1a77c15eba4bb9deacb8ee9a6987ff996169785ddfa14571b27e6a9267b5c5f4c768f8b2668023419f08c3b8cc73861f2d09c153d18b6fabd6422a95b7542fa3dc92c9'
>>> hex(phi)
'0xe4c1e6968dc199e487b3a1ac104c8922d86e75cd5ffbb83b9670974f3597231f0c97b7033e93e576524301df6ad19038c05eae26bfaea5ed1aaf48dc92b8ec8c4d6c2fd4269c3ea3e4af3008d0adcaa6692de680dd5b647328f3d39bc78a0521a3dc66c5dc22a3916231ea9a0cf3d6e75a03aa03c4901395ba0b32c7ecdd182006aae0842b38ff0c8c433c97432237cd7292494ba47185d156cd310168b0ea70a90aa5abc7f3ddd4c20b85f3da7921238d72dbcbf8a334541f3627eea4dad6cd64e0d6cb959125147eaabb82400c802da8f729c35645eb827860b0f4e630077e5ebe31342b83399fd4276fe6543abef62a08e41b16b689ef5a04b98233b50434c74dffe05a479d21ddd8192875c28c1d33501669b73e129fb43aabc96033c787511f00ab00dd97f3af432cc631b3b781edaa482cc26e835b520211cbbd6f34c7e1d3756eec46ec39a9a1ea830c50199929663aa1458df28f90f14bb7bf9455fa32343670c3dcd1741a291931b1cbc8077e36f969c0203f19fb9062175e3abdc'
>>> e*d % phi
1
```

Hence, our private and public key files check out.