RSA with dubious modulus

João Nuno da Silva Luís, NMEC 107403

Como a chave pública vinha num ficheiro .pem, começei por ir ao CyberChef e convertê-la para JWK, obtendo:

{“kty”:“RSA”,“n”:“gAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAOX2AAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAATKMk”,
“e”:“AQAB”}

O valor de **e** corresponde ao número inteiro 65537, e dando decode do **n** a partir de base 64, obtive o seguinte número:

n:
5221944407065762533458763553583121912899821245236918901921167416419769539857 78
7284244134059674987791704450533572196314189937867190928968036316180439256826 38
9729784882718549991701807950671918591572140350059279731131881594196988563728 36
1673421722933087484039543529018520356420243700593045572339888917990145033434 69
4884408938929734528150951304702997897267164117346515133482215295125079861999 33
8571077708469177799426457431591189572172483670439059363197482375500945206745 04
2085308375468341669252755164860441347753849918081847059665076068984129185940 45
9168283756106592464231840627751129991502061723924312978372460973086308096371 70
2527428999337171064188088055104373954832581618761742162375313391697885951182 71
1532072624130220686549205880932061675482060373385801371062403037187522633708 55
4529982725082753250016934789753562077600760620222688705296707631718544778308 51
0677289028160442959913924602957345163171702357286314780694306930477218438313 22
6180577650562067457668730546251124127523899241958464942323975784807854669350 53
3595084786079114668521561316962086747839793228012280923386686628060657249113 75
3296208083666036021308304544525099652288934293550232978774074357485602908695 91
32907771530539474239332906808611252830478647123436260345952071 3

Para fatorizar este número, tentei usar o GNFS, mas sem sucesso. Depois encontrei o site https://www.alpertron.com.ar/ECM.HTM, onde obtive a seguinte fatorização:

1

4039 625758 913875 912589 359586 083743 995055 512833 714435 504016 293178 440581 892358 486361
649650 176440 364182 961089 745115 237252 436764 944893 811365 136886 019048 306035 390078 859670
912624 511468 774718 798706 513349 507204 798008 446034 599027 330327 469520 229761 792309 521308
822705 731504 538130 360946 986442 633226 075944 049890 491298 190239 291673 708542 802585 621848
320571 186857 022004 415790 257259 725707 416378 274088 557539 268782 324108 022139 590742 950464
807732 169699 793894 037705 738050 462201 654160 903903 390710 588852 526215 644637 715866 415433
709817 822582 087241 880749 658544 124829 776940 645798 679666 942950 266929 153700 580664 809825
619018 524194 481701 382449 529707 (616 digits)
129268 024285 244029 202859 506754 679807 841776 410678 861936 128521 381710 098620 555471 563572
788805 646091 653854 754871 843687 592077 976478 236601 963684 380352 609545 793132 482523 509469
203984 367000 791001 558608 427184 230553 536270 273107 168874 570479 024647 352377 353904 681882
326583 408145 220171 550303 566164 263234 430209 596495 721542 087657 333558 673369 682739 899146
258277 979424 704141 305288 232311 222637 324104 770833 841256 601034 371456 708466 903774 414873
847429 430393 404609 206583 617614 790452 933148 924908 502738 843280 838900 628406 907725 293878
714170 322626 791740 183989 073411 994552 862100 665557 749342 174408 541732 918418 581273 914419
808592 774223 414444 238384 924059 (618 digits)
522 194440 706576 253345 876355 358312 191289 982124 523691 890192 116741 641976 953985 778728
424413 405967 498779 170445 053357 219631 418993 786719 092896 803631 618043 925682 638972 978488
271854 999170 180795 067191 859157 214035 005927 973113 188159 419698 856372 836167 342172 293308
748403 954352 901852 035642 024370 059304 557233 988891 799014 503343 469488 440893 892973 452815
095130 470299 789726 716411 734651 513348 221529 512507 986199 933857 107770 846917 779942 645743
159118 957217 248367 043905 936319 748237 550094 520674 504208 530837 546834 166925 275516 486044
134775 384991 808184 705966 507606 898412 918594 045916 828375 610659 246423 184062 775112 999150
206172 392431 297837 246097 308630 809637 170252 742899 933717 106418 808805 510437 395483 258161
876174 216237 531339 169788 595118 271153 207262 413022 068654 920588 093206 167548 206037 338580
137106 240303 718752 263370 855452 998272 508275 325001 693478 975356 207760 076062 022268 870529
670763 171854 477830 851067 728902 816044 295991 392460 295734 516317 170235 728631 478069 430693
047721 843831 322618 057765 056206 745766 873054 625112 412752 389924 195846 494232 397578 480785
466935 053359 508478 607911 466852 156131 696208 674783 979322 801228 092338 668662 806065 724911
375329 620808 366603 602130 830454 452509 965228 893429 355023 297877 407435 748560 290869 591329
077715 305394 742393 329068 086112 528304 786471 234362 603459 520713 (1233 digits)

E o seguinte totiente de Euler:

Euler's totient: 522 194440 706576 253345 876355 358312 191289 982124 523691 890192 116741 641976 953985
778728 424413 405967 498779 170445 053357 219631 418993 786719 092896 803631 618043 925682 638972
978488 271854 999170 180795 067191 859157 214035 005927 973113 188159 419698 856372 836167 342172
293308 748403 954352 901852 035642 024370 059304 557233 988891 799014 503343 469488 440893 892973
452815 095130 470299 789726 716411 734651 513348 221529 512507 986199 933857 107770 846917 779942
645743 159118 957217 248367 043905 936319 748237 550094 520674 504208 530837 546834 166925 275516
486044 134775 384991 808184 705966 507606 898412 918594 045916 828375 610659 246423 184062 775112
999150 206172 392431 297837 246097 308630 676329 520208 584994 818268 240078 045253 673605 471970
681790 243636 541348 992136 721958 545183 832697 384730 394984 352693 331785 263875 754305 024541
563530 619867 611709 619584 390768 486312 881663 629805 759281 336164 034822 470001 741783 303127
102627 769956 677686 895691 704853 525711 666755 156341 634158 384483 963710 273775 222477 831683
217852 769825 218598 940705 572439 535212 166917 706772 898966 691673 900353 247501 753749 352656
081989 597118 357794 777872 113394 101513 500970 096115 476280 735033 445562 839684 081352 977253
831461 943196 255692 093558 978539 121142 028521 820020 014447 290284 565921 288053 109518 979513
162319 960356 496732 670274 167129 361867 100693 488053 338216 982625 066948 (1233 digits)

Com todos estes valores, era agora possível encontrar o valor do expoente **d**. Para isso, desenolvi o seguinte código:

calculate_d.py > ...

```python
def calculate_d(totient, e):
    """    Calculate the private key (d) using the Extended Euclidean Algorithm
    d * e ≡ 1 (mod φ(n))
    """
    def extended_gcd(a, b):
        if a == 0:
            return b, 0, 1
        gcd, x1, y1 = extended_gcd(b % a, a)
        x = y1 - (b // a) * x1
        y = x1
        return gcd, x, y

    # Calculate d using Extended Euclidean Algorithm
    gcd, x, y = extended_gcd(e, totient)

    # Make sure d is positive
    d = x % totient
    if d < 0:
        d += totient

    return d

# Example usage
def main():
    # Example values
    a= "4039 625758 913875 912589 359586 083743 995055 512833 714435 504016 293178 440581 892358 486361 649650 176440 364182 961089 745115 237252 436764 944893 811365 136886 019048 306035 390078 859670 912624 511468 774718 798706
    a_certo= a.replace(" ", "")

    b= "129268 024285 244029 202859 506754 679807 841776 410678 861936 128521 381710 098620 555471 563572 788805 646091 653854 754871 843687 592077 976478 236601 963684 380352 609545 793132 482523 509469 203984 367000 791001 5586
    b_certo= b.replace(" ", "")

    euler_phi= "522 194440 706576 253345 876355 358312 191289 982124 523691 890192 116741 641976 953985 778728 424413 405967 498779 170445 053357 219631 418993 786719 092896 803631 618043 925682 638972 978488 271854 999170 180795
    totient= euler_phi.replace(" ", "")

    e= 65537

    # Calculate private key
    d = calculate_d(int(totient), e)

    print(f"Private key d = {d}")

    # Verify that (d * e) mod totient = 1
    verification = (d * e) % int(totient)
    print(f"Verification (should be 1): {verification}")

if __name__ == "__main__":
    main()
```

E obtive o seguinte número:

d:
270670692140354232359726868662310834156593874758835673128556475175518518194255
205526303056299738064428033301227470755135316217325290838830269406059968497783
632331038140209565921708982840723674497925763601497982004898023642937121793570
725004403509524363081653560096982065867518620793056987796795743693066858088982
689508783825080613884199483987306160749142568421290445220853645384132570780044
144925019083415429217871979112795382404899934822794523044720503373311425108151
242256931984466128300831733143581782173731299444955284216278795281124965204994
732664905618110298014794125646132544686703750189524866678841721860539604725785
456850821276173338960452832251894012280758356570736123252899322282755264658052
653160188771497298640347906456106617235175300622749517934949261879202309449707
494218382188755994976684765739397886780889701677025614176195819588034604065599
796668575389774824311843879950576319441506326873484778709414688780245703584326
651636415627983540750942201739138174102604559253820507116064356426220098785580
882381010966202261751585027605246543620497872745865231305495270772804868315031
362082577176529265925720511996412503258920774134473148059052980211641664007539
618979405682194925144780160587494475037883778200088971112097353