Configuration Manual

Research Project
MSc in Cybersecurity

Olwen Brangan
Student ID: x20216866

School of Computing
National College of Ireland

Supervisor: Ross Spelman

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Olwen Brangan………………………………………………………………………………
Student ID:	……………………………x20216866……………………………………………………………
Programme:	…MSc Cybersecurity……………………………… **Year:** …2024………………………..
Module:	Research Project…………………………………………………………………………………………………
Lecturer:	………Ross Spelman……………………………………………………………………………
Submission Due Date:	12 August 2024 ………………………………………………………………………………………
Project Title:	Time efficient factorization of RSA semiprime numbers ……………………………………………………
Word Count:	……………………………………… **Page Count:** …………………………………………

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: …Olwen Brangan………………………………………………………………………………

Date: 12 August 2024………………………………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□

You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Olwen Brangan
Student ID: x20216866

1 Requirements Specification

1. Is this research ethical?
2. The research should be based on a genuine problem that exists.
3. The research should be unique.
4. The research should be based on an ICT solution.
5. The solution should be measurable.
6. Software that can perform large computations and be used in the OT industry.
7. The solution should make a significant contribution to research. Who can benefit from this research?.
8. An interdisciplinary project is required.
9. The process from collecting data to analysing data and obtaining results should be clearly explained.
10. The research method used should be easily reproduced by other researchers so the results can be verified.
11. Create a unique algorithm.
12. Method for factorizing large semiprime numbers. The semi-prime number to be tested should be at least 100 digits.
13. Verify that factors obtained are prime numbers.
14. Use a single computer for all tasks.

Figure 1: List of requirements

2 Data organisation

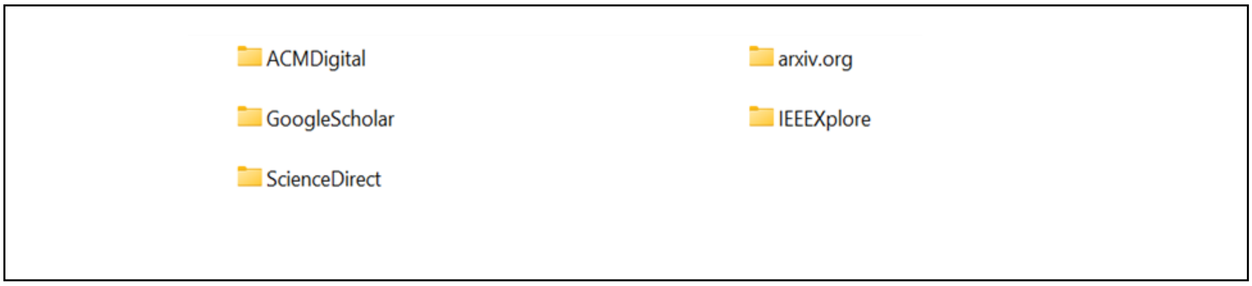

Figure 2: Organising data

3. Mind map of measurements

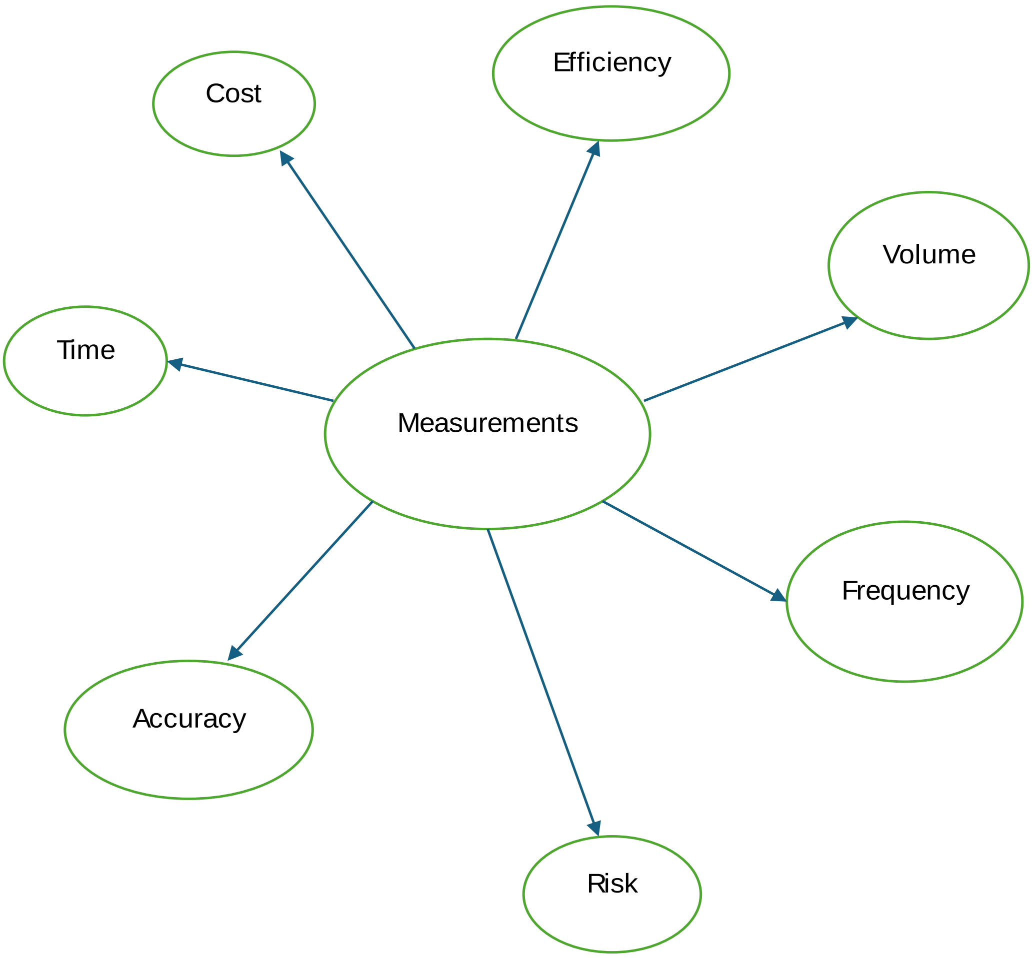

Figure 3 : Mind map for generating ideas of what can be measured.

4 The RSA Algorithm

4.1 Generating Public and Private Keys

The steps involved in generating the public and private keys used in the RSA cryptographic scheme are as follows:

A. Select two prime numbers and name these p and q.
B. Calculate N (p x q).
C. Calculate Φ(n). Φ(n) = (p-1) (q-1).
D. Select the encryption/public exponent key such that the greatest common denominator - G.C.D. (e, Φ(n)) = 1.
E. Calculate the private/decryption key d such that d x e ≡ 1 mod Φ(n).

Example:

A. let p = 11 and q = 13
B. N = 11 X 13 = 143
C. Φ(n) = (p-1) (q-1) = 10 x 12 = 120
D. Let e = 7. (G.C.D 7, 120 = 1)
E. de = 1 mod 120. 120/7 = 17. d = 17
Proof: 17 x 7 = 119. 119 / 120 = 1

4.2 Digital Signatures

1. Generate public and private keys by following the steps described in step 4.1
2. Assign a value to message x.
3. Calculate signature s using the following formula: $s = x^d \bmod n$.

Example:

1. A. let p = 3 and q = 11
B. N = 3 X 11 = 33
C. Φ(n) = (p-1) (q-1) = 2 x 10 = 20
D. Let e = 3. (G.C.D 3, 20 = 1)
E. de = 1 mod 20. 20/3 = 7. d = 7
Proof: 7 x 3 = 21. 21 / 20 = 1
2. Let message x = 2
3. $s = x^d \bmod n = 2^7 \bmod 33 = 128 \bmod 33$

4.3 Hamming weight

Hamming weight is the number of 1's in a binary string.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2^{16}	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
65536	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Table 1: Binary system

The closer a prime number is to one of the numbers in the table above, the lower the hamming weight. For example, to determine the hamming weight of prime number 17, select 16 in the table above. This corresponds to one binary 1 in the string. An additional 1 is required in the binary string to represent the number 17. This gives 17 a Hamming Weight of two.

A quick way to work out the numbers that are more likely than other numbers to have low hamming weight is as follows:

1. Select a number: for example, 128. 128 is written in binary as 1 0 0 0 0 0 0 0:

2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
1	0	0	0	0	0	0	0
128 x 1	64 x 0	32 x 0	16 x 0	8 x 0	4 x 0	2 x 0	1 x 0

Table 2: Binary table from 0 to 128

2. Find the next prime that is greater than 128. The next prime is 131.
3. Subtract 128 from 131. The remainder is three.
4. Check the chart for a number corresponding to 3. 2^0 (1) plus 2^1 (2) gives a total of three.

2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
1	0	0	0	0	0	1	1
128 x 1	64 x 0	32 x 0	16 x 0	8 x 0	4 x 0	2 x 1	1 x 1

Table 3: Binary representation for decimal number 131

5. The number 131 is written in binary as 1 0 0 0 0 0 1 1. Two additional 1's are required in the binary string. Therefore three 1's are required so the Hamming Weight for 131 is three.

4.4 The choice of the public exponent e

Popular choices of e include the numbers 3 and 65537 (2^{16} + 1). These are prime numbers with a Hamming weight of two. Using the procedure described above, I found that 17 and 257 also have a Hamming weight of two. 65537 is the largest number and therefore more secure than 3, 17 and 257. Using e with a low hamming weight means there are less computations required for encryption which results in faster encryption. Using e equal to 3, 17, 257 or 65537 allows for fast encryption, though 65537 is the best option for security purposes. The number of calculations required for e is explained below:

Example:

$e = 3$

$8^3 = 8 \text{ X } 8 \text{ X } 8 = 512$

$8^2 = 64 \text{ X } 8 = 512$. 8 is squared once and the result is multiplied by 8, therefore, to speed up exponentiation there are two calculations required: one squaring and one multiplication.

Example:

$e = 17$

$2^{17} = 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 \text{ x } 2 = 131072$

2^{17}: $2^2 = 4$,
$4^2 = 16$,
$16^2 = 256$,
$256^2 = 65536$,
$65536 \text{ x } 2 = 131072$

Therefore, when e is equal to 17 only five calculations are required: four squaring operations and one multiplication.

5 Software

R Studio	Python	SageMath
Free, open-source software	Free, open source software	Free, open source software. Builds on top of R, Python and other open-source packages.
Runs on different operating systems	Runs on different operating systems. Unofficial builds available for Android and Ios.	Can be used anywhere. Has an easy to use web interface SageMathCell which is extremely useful for checking if a number is prime or not.
Useful for displaying table of prime numbers, statistical analysis and analysing small numbers. More complex when dealing with large numbers.	Useful for calculations of large numbers	Extremely poweful math calculator
Bigz which is used for storing large numbers does not retain complete accuracy when returning a result.	Decimal module allows 100% accuracy when dealing with large number. If the same calculation is run multiple times, the same result is obtained. This means it has also 100% precision.	Can handle numbers containing millions of digits

Table 4: A Comparison of RStudio, Python and SageMath

5.1 R Studio

5.1.1 R Studio version and build

This can be found by clicking on the help menu and selecting the About Rstudio option:

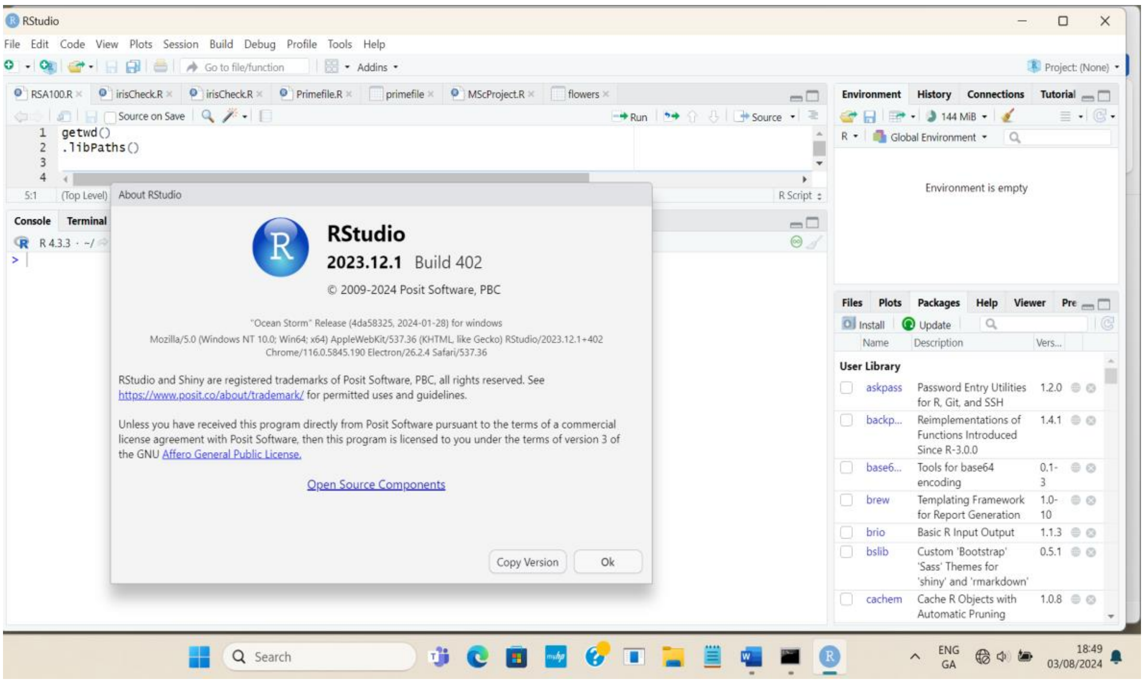

Figure 4: R Studio

5.1.2 R Studio command to generate prime numbers from 1 to 5000.

R command: generate_primes(1,5000)

1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83
89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193
197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311
313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433
439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563 569
571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677 683
691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809 811 821 823 827
829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971
977 983 991 997 1009 1013 1019 1021 1031 1033 1039 1049 1051 1061 1063 1069 1087 1091
1093 1097 1103 1109 1117 1123 1129 1151 1153 1163 1171 1181 1187 1193 1201 1213 1217 1223
1229 1231 1237 1249 1259 1277 1279 1283 1289 1291 1297 1301 1303 1307 1319 1321 1327 1361
1367 1373 1381 1399 1409 1423 1427 1429 1433 1439 1447 1451 1453 1459 1471 1481 1483 1487
1489 1493 1499 1511 1523 1531 1543 1549 1553 1559 1567 1571 1579 1583 1597 1601 1607 1609
1613 1619 1621 1627 1637 1657 1663 1667 1669 1693 1697 1699 1709 1721 1723 1733 1741 1747
1753 1759 1777 1783 1787 1789 1801 1811 1823 1831 1847 1861 1867 1871 1873 1877 1879 1889
1901 1907 1913 1931 1933 1949 1951 1973 1979 1987 1993 1997 1999 2003 2011 2017 2027 2029
2039 2053 2063 2069 2081 2083 2087 2089 2099 2111 2113 2129 2131 2137 2141 2143 2153 2161
2179 2203 2207 2213 2221 2237 2239 2243 2251 2267 2269 2273 2281 2287 2293 2297 2309 2311
2333 2339 2341 2347 2351 2357 2371 2377 2381 2383 2389 2393 2399 2411 2417 2423 2437 2441
2447 2459 2467 2473 2477 2503 2521 2531 2539 2543 2549 2551 2557 2579 2591 2593 2609 2617
2621 2633 2647 2657 2659 2663 2671 2677 2683 2687 2689 2693 2699 2707 2711 2713 2719 2729
2731 2741 2749 2753 2767 2777 2789 2791 2797 2801 2803 2819 2833 2837 2843 2851 2857 2861
2879 2887 2897 2903 2909 2917 2927 2939 2953 2957 2963 2969 2971 2999 3001 3011 3019 3023
3037 3041 3049 3061 3067 3079 3083 3089 3109 3119 3121 3137 3163 3167 3169 3181 3187 3191
3203 3209 3217 3221 3229 3251 3253 3257 3259 3271 3299 3301 3307 3313 3319 3323 3329 3331
3343 3347 3359 3361 3371 3373 3389 3391 3407 3413 3433 3449 3457 3461 3463 3467 3469 3491
3499 3511 3517 3527 3529 3533 3539 3541 3547 3557 3559 3571 3581 3583 3593 3607 3613 3617
3623 3631 3637 3643 3659 3671 3673 3677 3691 3697 3701 3709 3719 3727 3733 3739 3761 3767
3769 3779 3793 3797 3803 3821 3823 3833 3847 3851 3853 3863 3877 3881 3889 3907 3911 3917
3919 3923 3929 3931 3943 3947 3967 3989 4001 4003 4007 4013 4019 4021 4027 4049 4051 4057
4073 4079 4091 4093 4099 4111 4127 4129 4133 4139 4153 4157 4159 4177 4201 4211 4217 4219
4229 4231 4241 4243 4253 4259 4261 4271 4273 4283 4289 4297 4327 4337 4339 4349 4357 4363
4373 4391 4397 4409 4421 4423 4441 4447 4451 4457 4463 4481 4483 4493 4507 4513 4517 4519
4523 4547 4549 4561 4567 4583 4591 4597 4603 4621 4637 4639 4643 4649 4651 4657 4663 4673
4679 4691 4703 4721 4723 4729 4733 4751 4759 4783 4787 4789 4793 4799 4801 4813 4817 4831
4861 4871 4877 4889 4903 4909 4919 4931 4933 4937 4943 4951 4957 4967 4969 4973 4987 4993
4999

Figure 5: Prime numbers from 1 to 5000

5.1.3 Observations about prime numbers

1. Any even greater than two cannot be a prime number.
2. Semi-prime numbers have two prime numbers other than itself and one.
3. The numbers 21 is an odd number ending in one with two prime numbers other than itself and one (3 x 7 = 21).
4. The numbers 33 is an odd number ending in three with two prime numbers other than itself and one (11 x 3 = 33).
5. The following numbers are odd numbers ending in five with **two** prime numbers other than itself and one: 15, 25, 35, 55, 65, 85 and 95. (3 x 5 = 15), (7 x 5 = 35), (11 x 5 = 55), (13 x 5 = 65), (17 x 5 = 85) and (19 x 5 = 95).
6. The number 77 is an odd number ending in seven with two prime numbers other than itself and one (11 x 7 = 77).
7. The following are odd numbers ending in nine with two prime numbers other than itself and one: 9, 39, 49 and 69 (3 x 3 = 9), (13 x 3 = 39), (7 x 7 = 49), (23 x 3 = 69).

5.1.4 Commands input in R studio to determine if 400 is a prime number and to find the next number greater than 400 that is a prime number:

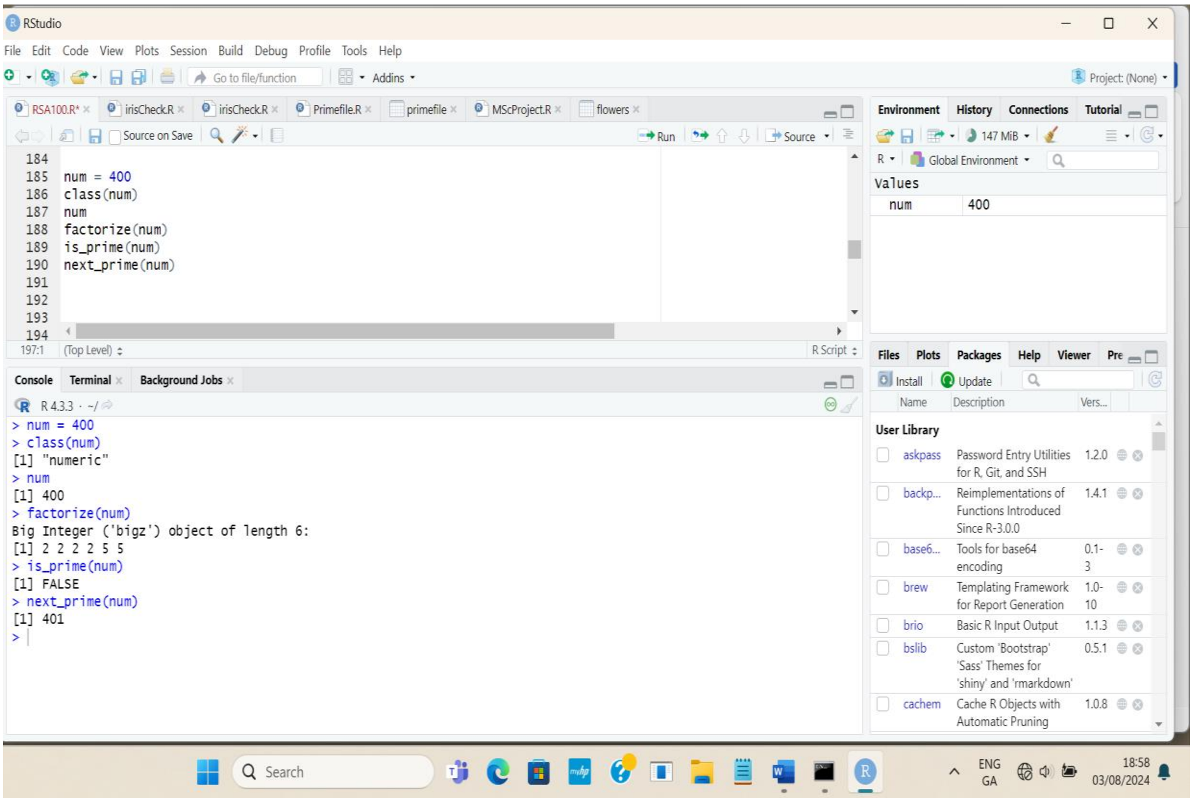

Figure 6: Check if 400 is a prime number and determine next prime after 400.

5.1.5 Factorising small numbers in R:

```
Console   Terminal ×   Background Jobs ×
R 4.3.3 · ~/
> num = 9
> factorize(num)
Big Integer ('bigz') object of length 2:
[1] 3 3
> num2 = 379
> factorize(num2)
Big Integer ('bigz') :
[1] 379
> num3 = 400946
> factorize(num3)
Big Integer ('bigz') object of length 4:
[1] 2     7     13    2203
> factorize(num3)
Big Integer ('bigz') object of length 4:
[1] 2     7     13    2203
```

Figure 7: Factorising numbers in R

5.1.6 Issues with R

Problems arose when dealing with larger numbers and calculations which result in a number generally greater than 15 digits. I found that when a 50-digit number is stored in a variable and the variable contents are checked the number returned is a shortened version of the number with scientific notation e+49 (Figure 8).

```
num4 = 37975227936943673922808872755445627854565536638199
class(num4)
num4
factorize(num4)
is_prime(num4)
next_prime(num4)

223:1   (Top Level)
```

```
Console   Terminal ×   Background Jobs ×
R 4.3.3 · ~/
> num4 = 37975227936943673922808872755445627854565536638199
> class(num4)
[1] "numeric"
> num4
[1] 3.797523e+49
```

Figure 8: A 50-digit number is displayed as scientific notation

However, scientific notation can be turned off using the command option scipen = 999 (Figure 9). However, note the loss of accuracy after the first 16 digits. To overcome issues with large number calculations, bigz is used which is obtained from the gmp package. However, the result returned from multiplying two numbers loses accuracy. The first fifteen digits are correct, but the digits vary from the original number after that.

```
Console   Terminal ×   Background Jobs ×
R 4.3.3 · ~/
> options(scipen = 999)
> num4 = as.bigz(37975227936943673922808872755445627854565536638199)
> num4
Big Integer ('bigz') :
[1] 37975227936943676397215001776962205505768793309184
```

Figure 9: Turning off scientific notation

5.2 Python

5.2.1 Checking the accuracy of Python

A series of calculations were performed to test the accuracy of Python:

1. Math module

```
>>> import math
>>> a = 1522605027922533360535618378132637429718068114961380688657908494580122963258952897654000350692006139
>>> b = 37975227936943673922808872755445627854565536638199
>>> c = 40094690950920881030683735292761468389214899724061
>>> b * c
1522605027922533360535618378132637429718068114961380688657908494580122963258952897654000350692006139
```

2. Mpmath module

```
>>> d = mpmath.sqrt(a)
>>> d
mpf('39020571855401265512289573339484371018905006900194.784438069009729506566899414351035827216720849279640784 9')
>>>
>>> d * d
mpf('1522605027922533360535618378132637429718068114961380688657908494580122963258952897654000350692006139.0')
```

3. However, when dividing 3 by 7, the result is a number which 17 digits after the decimal point.

```
>>> 3 / 7
0.42857142857142855
```

```
>>> d = 3
>>> e = 7
>>> d / e
0.42857142857142855
```

To obtain 50 digits, set the precision and to ensure an accurate result use the Decimal Module:

```
>>> 1 / 7
0.14285714285714285
>>> import decimal
>>> from decimal import Decimal, getcontext
>>> getcontext().prec = 50
>>> 1 / 7
0.14285714285714285
>>> ans = Decimal('1') / Decimal('7')
>>> ans
Decimal('0.14285714285714285714285714285714285714285714285714')
```

4. Setting number of decimal places using mpmath.mp.dps:

```
>>> import mpmath
>>> rsa100 = 1522605027922533360535618378132637429718068114961380688657908494580122963258952897654000350692006139
>>> p100 = 37975227936943673922808872755445627854565536638199
>>> q100 = 40094690950920881030683735292761468389214899724061
>>> sqrtrsa100 = mpmath.sqrt(rsa100)
>>> sqrtrsa100
mpf('3.9020571855401266e+49')
>>>
>>> mpmath.mp.dps = 102
>>> sqrtrsa100 = mpmath.sqrt(rsa100)
>>> sqrtrsa100
mpf('39020571855401265512289573339484371018905006900194.784438069009729506566899414351035827216720849279640 7849')
```

5.2.2 Using logarithms

The following number is the log of RSA-100 semi-prime number:

```
99.182587259580113217432403849517753671832055836240270307008869918397121676165817618356524006573854 7406
```

The process of using logarithms turns multiplication into addition. So instead of obtaining the square root as the maximum for one of the factors, the number can be divided by two to set the maximum standard.

```
>>> mean
Decimal('49.591293629790056608716201924758876835916027918120135153504434959198560838082908809178262003286927 3703
```

This returns a result of forty-nine point five. This is assigned to variable a. The initial two digits of p and q are forty-nine. Forty-nine multiplied by two results in ninety-eight. Therefore, there is an outstanding one to account for.

p – 4 9
q – 4 9
n – 9 9. 1 8

Because one is carried over and added to the second digit in p and q, it means that the third digit in p and q combined exceed ten. As the third digit in the semiprime number n is one, it means that the total of the third digit in p and the third digit in q is eleven, assuming there are no digits carried over from adding the fourth digit in p with the fourth digit in q.

Let p be the factor below N divided by two (forty-nine point five nine). Therefore, p must start with forty-nine point five nine or less. This means that the third digit of p must be five or less. If the third digit of p is five, the third digit of q must be six.

p – 4 9. 5
q – 4 9. 6
n – 9 9. 1 8 2 5 8 7 2 5 9 5

To obtain the fourth digit of p and q, take the mean and minus one for p. Add one for q as one value lies above the mean. Now p equals forty-nine point five eight and q equals forty-nine point sixty. Adding these two numbers, results in exactly one hundred and eighteen. As there are more digits in both numbers the first four digits should be less than one hundred and eighteen. Change the fourth digit in p to seven.

p – 4 9. 5 7
q – 4 9. 6 0
n – 9 9. 1 8 2 5 8 7 2 5 9 5

Add the highest possible value to p and q (nine) which now contain four digits each. Calculate the inverse log of p and the inverse log of q. Multiply the results together. This gives the potential value of N. The table below shows that the result exceeds the first four digits of N (1522).

p	Inverse log p	q	Inverse log q	inverse log p * q
49.579	3.793×10^{49}	49.609	4.064×10^{49}	1.542×10^{99}

Keep the highest value of p which is nine. As the fifth digit in N is 2, q must be three. Now the initial five digits of p and q are:

p – 4 9. 5 7 9
q – 4 9. 6 0 3
n – 9 9. 1 8 2 5 8 7 2 5 9 5

5.3 Wolfram Alfra

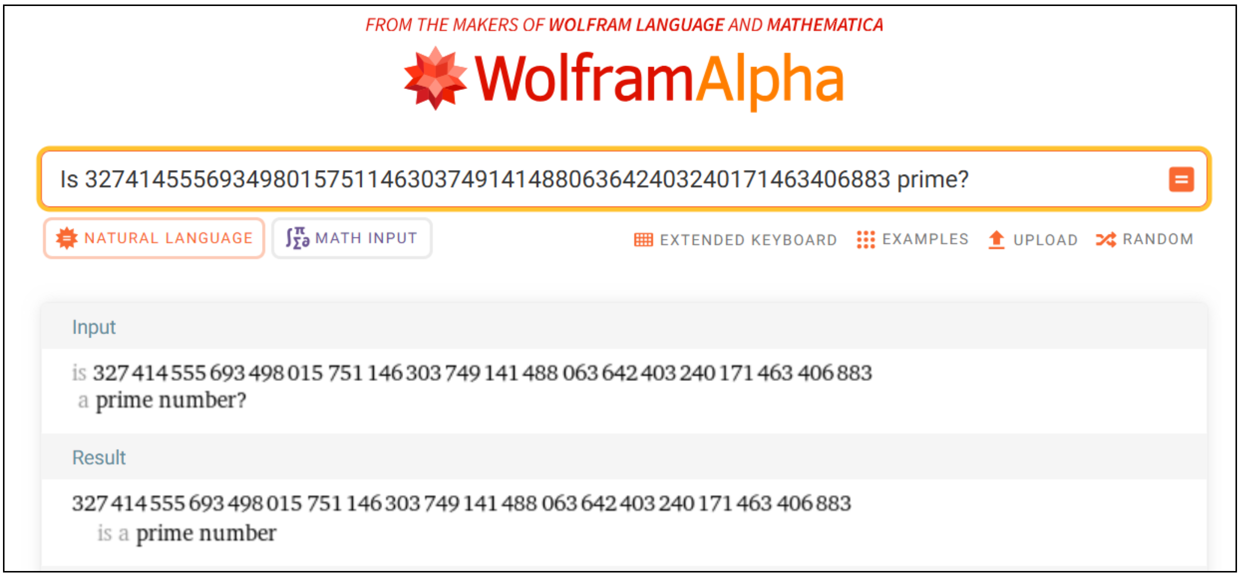

Figure 10: Wolfram Alpha Software Tool for checking for prime numbers

5.4 SageMath

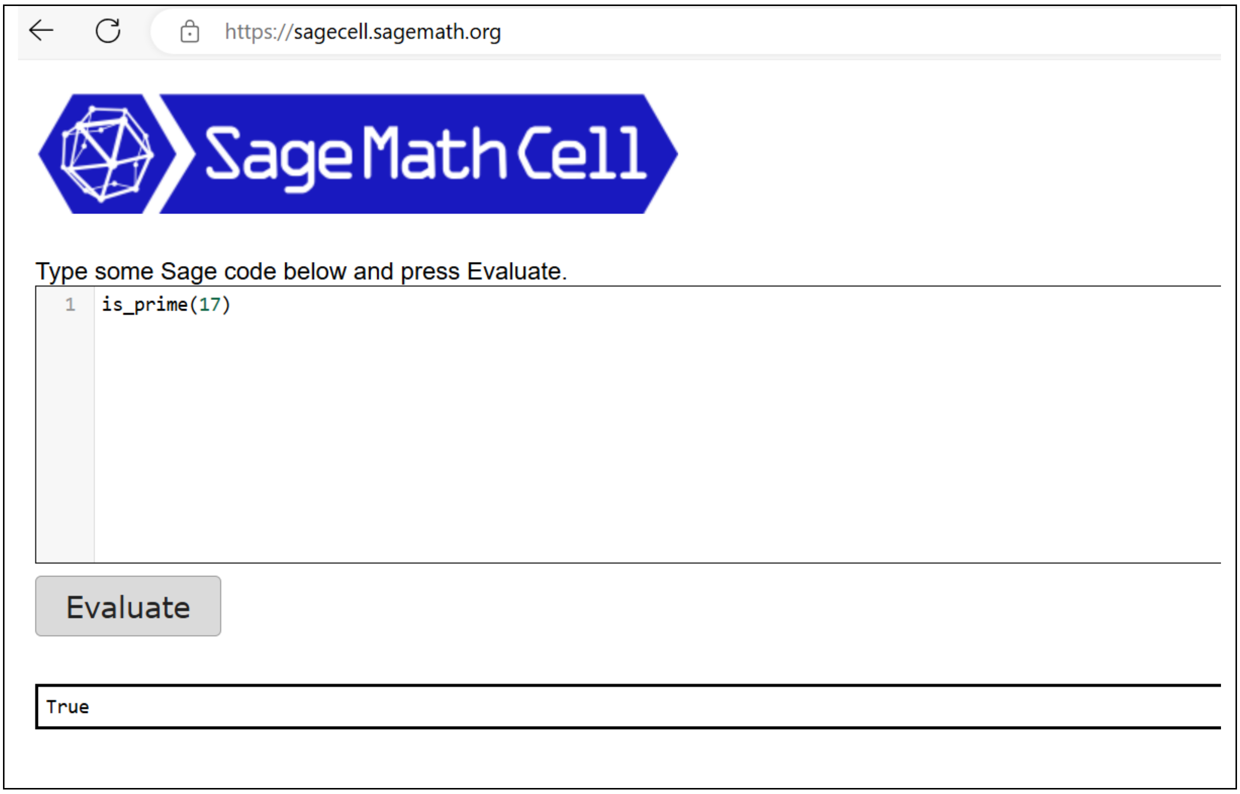

Figure 11: SageMath software tool for checking primality of numbers

5.5 Calculator.net

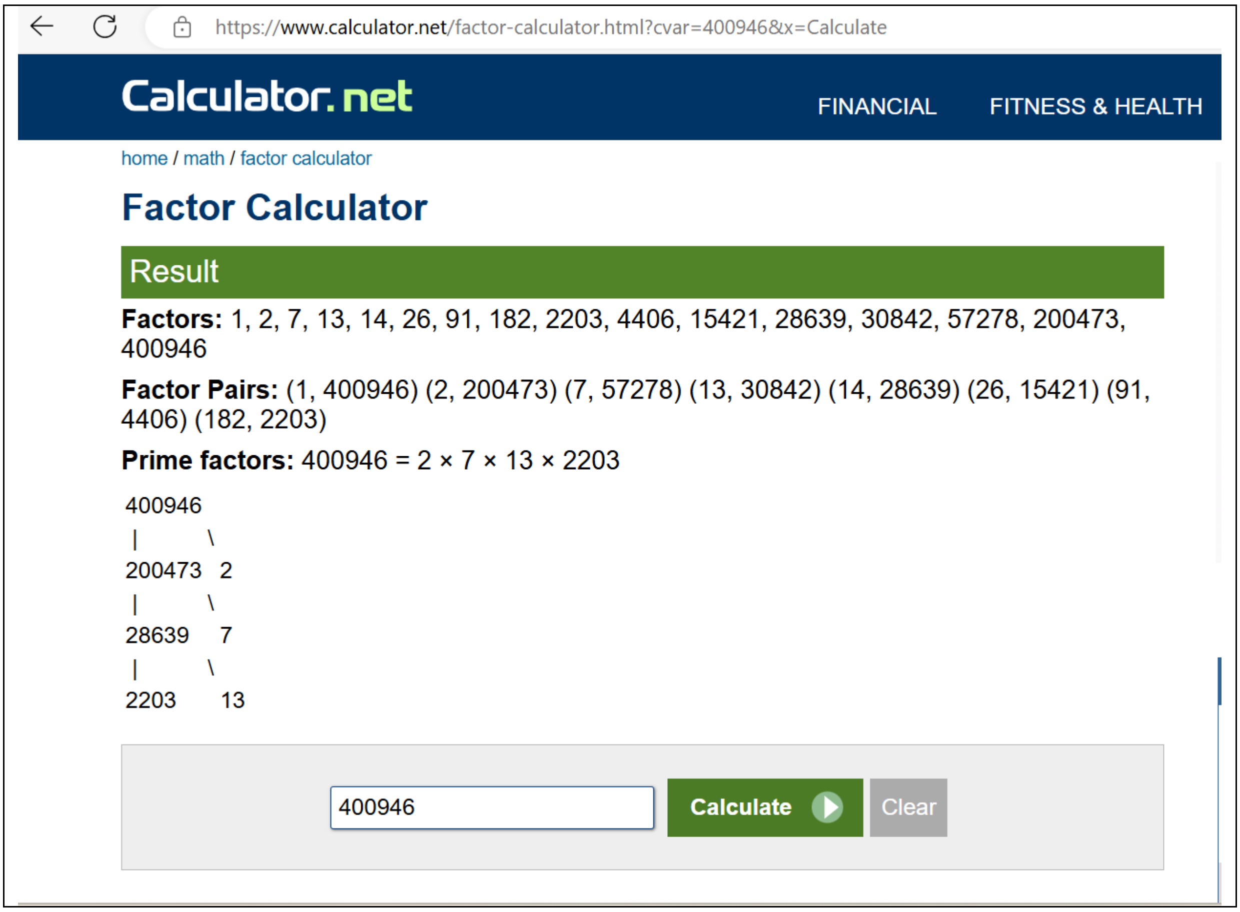

Figure 12: Online Calculator

Explanation of terms

Accuracy
How close a measurement is to its actual value

Composite number
A number with more than two factors.

Coprime
Two numbers that have no common factors (Eight and nine. The factors of eight are 1, 2 4 and 8 whilst the factors of 9 are 1, 3 and 9)

Exponentiation
How many times a number is multiplied by itself.

Factor
Whole numbers that divide evenly into a given number.

Integer
Positive or negative whole number including zero. e.g. {-4. -3. -2. -1. 0, 1, 2, 3, 4}

Irrational number
Decimals that never repeat or end e.g. pi

Method
A step-by-step explanation of what to do to achieve an outcome. For example, the steps required in Python to run a particular algorithm to factorise a large number. A method explains how to do something, not why particular steps are required.

Methodology
The approach taken e.g. the design and rationale for choosing a particular method. For example, the process of selecting suitable software for factorising large semi-prime numbers. This includes testing different software to see what is most suitable, checking the literature to see what software other people have used, testing different software to verify accuracy of claims about software from other people, considering the requirements of this project e.g. software which can easily be used anywhere, on various operating systems, easy to implement, cheap and suitable for use in the operational technology industry.

Natural number
Whole numbers greater than zero {1, 2, 3, 4, 5}

Polynomial factorisation
The number of steps required to factorise N increases at a predictable rate.

Precision
Obtaining the same results when a test is performed multiple times.

Prime number
A whole number greater than one that has only two factors, itself and one.

6 Useful mathematical formulae

$e = 2.71828$

$\text{Log}(A \times B) = \text{Log } A + \text{Log } B$

References

RStudio Team (2020). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA URL http://www.rstudio.com/.

Easttom, W. (2021). *Modern cryptography: Applied mathematics for encryption and information security.* Cham, Switzerland. Springer.

sagecell.sagemath.org. (n.d.). *Sage Cell Server*. [online] Available at: https://sagecell.sagemath.org/.

Wolfram Alpha (2024). *Wolfram|Alpha: Making the world's knowledge computable*. [online] Wolframalpha.com. Available at: https://www.wolframalpha.com/.

Python (2019). *Download Python*. [online] Python.org. Available at: https://www.python.org/downloads/.

RSA Digital Risk Management & Cyber Security Solutions (2019). RSA Digital Risk Management & Cyber Security Solutions. [online] RSA.com. Available at: https://www.rsa.com/.

APPENDIX A

Search terms

ACM Digital Library

Advanced Search:
The ACM Full-Text collection
Search Within = Anywhere
Search term = Rivest AND Shamir AND Adleman
Publication Date = Last 5 years
Number of results returned = 165

William Gasarch. 2021. Review of Ideas that Created the Future: Classic Papers of Computer Science Edited by Harry Lewis. SIGACT News 52, 2 (June 2021), 10–17. https://doi.org/10.1145/3471469.3471473

Advanced Search:
The ACM Full-Text collection
Search Within = Anywhere
Search term = RSA AND prime
Publication Date = Last 5 years
Number of results returned = 102

102 Results for: [[All: and] OR [[All: rivest] AND [All: shamir] AND [All: adleman]]] AND [All: rsa] AND [All: prime] AND [E-Publication Date: Past 5 years]

Sandeep Joshi, Amit Kumar Bairwa, Anton Pavlovich Pljonkin, Pradumn Garg, and Kshitij Agrawal. 2023. From Pre-Quantum to Post-Quantum RSA. In Proceedings of the 6th International Conference on Networking, Intelligent Systems & Security (NISS '23). Association for Computing Machinery, New York, NY, USA, Article 1, 1–8. https://doi.org/10.1145/3607720.3607721

Abdelrahman Abuarqoub, Simak Abuarqoub, Ahmad Alzu'bi, and Ammar Muthanna. 2022. The Impact of Quantum Computing on Security in Emerging Technologies. In The 5th International Conference on Future Networks & Distributed Systems (ICFNDS 2021). Association for Computing Machinery, New York, NY, USA, 171–176. https://doi.org/10.1145/3508072.3508099

2023. Proceedings of the 2023 ACM Southeast Conference. Association for Computing Machinery, New York, NY, USA.
2023. Companion of the 19th International Conference on emerging Networking EXperiments and Technologies. Association for Computing Machinery, New York, NY, USA.

2023. Proceedings of the 18th International Conference on Availability, Reliability and Security. Association for Computing Machinery, New York, NY, USA.

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = RSA AND prime
Publication Date = Last 5 years
Number of results returned =
112,934 Results for: [All: rsa and prime] AND [E-Publication Date: Past 5 years]

Xiaona Zhang, Yang Liu, and Yu Chen. 2021. Attack on the Common Prime Version of Murru and Saettone’s RSA Cryptosystem. In Innovative Security Solutions for Information Technology and Communications: 14th International Conference, SecITC 2021, Virtual Event, November 25–26, 2021, Revised Selected Papers. Springer-Verlag, Berlin, Heidelberg, 32–45. https://doi.org/10.1007/978-3-031-17510-7_3

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = RSA AND prime
Publication Date = Last 5 years
Number of results returned =
112,934 Results for: [All: rsa and prime] AND [E-Publication Date: Past 5 years]

Aykan Inan. 2022. Method for Approximating RSA Prime Factors. In Proceedings of the 2022 European Interdisciplinary Cybersecurity Conference (EICC '22). Association for Computing Machinery, New York, NY, USA, 1–5. https://doi.org/10.1145/3528580.3528581

Andrey Ivanov and Nikolai Stoianov. 2023. Implications of the Arithmetic Ratio of Prime Numbers for RSA Security. Int. J. Appl. Math. Comput. Sci. 33, 1 (Mar 2023), 57–70. https://doi.org/10.34768/amcs-2023-0005

Meryem Cherkaoui-Semmouni, Abderrahmane Nitaj, Willy Susilo, and Joseph Tonien. 2021. Cryptanalysis of RSA Variants with Primes Sharing Most Significant Bits. In Information Security: 24th International Conference, ISC 2021, Virtual Event, November 10–12, 2021, Proceedings. Springer-Verlag, Berlin, Heidelberg, 42–53. https://doi.org/10.1007/978-3-030-91356-4_3

Kyuchol Kim, Yongbok Jong, and Yunmi Song. 2024. Decryption speed up of RSA by pre-calculation. In Proceedings of the 2023 International Conference on Mathematics, Intelligent Computing and Machine Learning (MICML '23). Association for Computing Machinery, New York, NY, USA, 11–16. https://doi.org/10.1145/3638264.3638269

Wan Nur Aqlili Wan Mohd Ruzai, Abderrahmane Nitaj, Muhammad Rezal Kamel Ariffin, Zahari Mahad, and Muhammad Asyraf Asbullah. 2022. Increment of insecure RSA private exponent bound through perfect square RSA diophantine parameters cryptanalysis. Comput. Stand. Interfaces eighty, C (Mar 2022). https://doi.org/10.1016/j.csi.2021.103584

Abderrahmane Nitaj, Willy Susilo, and Joseph Tonien. 2023. A new attack on some RSA variants. Theor. Comput. Sci. 960, C (Jun 2023). https://doi.org/10.1016/j.tcs.2023.113898

Abderrahmane Nitaj, Muhammad Rezal Bin Kamel Ariffin, Nurul Nur Hanisah Adenan, and Nur Azman Abu. 2021. Classical Attacks on a Variant of the RSA Cryptosystem. In Progress in Cryptology – LATINCRYPT 2021: 7th International Conference on Cryptology and Information Security in Latin America, Bogotá, Colombia, October 6–8, 2021, Proceedings. Springer-Verlag, Berlin, Heidelberg, 151–167. https://doi.org/10.1007/978-3-030-88238-9_8

Abderrahmane Nitaj and Maher Boudabra. 2023. Improved Cryptanalysis of the Multi-Power RSA Cryptosystem Variant. In Progress in Cryptology - AFRICACRYPT 2023: 14th International Conference on Cryptology in Africa, Sousse, Tunisia, July 19–21, 2023, Proceedings. Springer-Verlag, Berlin, Heidelberg, 252–269. https://doi.org/10.1007/978-3-031-37679-5_11

Mengce Zheng and Honggang Hu. 2019. Implicit Related-Key Factorization Problem on the RSA Cryptosystem. In Cryptology and Network Security: 18th International Conference, CANS 2019, Fuzhou, China, October 25–27, 2019, Proceedings. Springer-Verlag, Berlin, Heidelberg, 525–537. https://doi.org/10.1007/978-3-030-31578-8_29

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = RSA AND decryption
Publication Date = Last 5 years
Number of results returned =
87,593 Results for: [All: rsa decryption] AND [E-Publication Date: Past 5 years]

Kyuchol Kim, Yongbok Jong, and Yunmi Song. 2024. Decryption speed up of RSA by pre-calculation. In Proceedings of the 2023 International Conference on Mathematics, Intelligent Computing and Machine Learning (MICML '23). Association for Computing Machinery, New York, NY, USA, 11–16. https://doi.org/10.1145/3638264.3638269

Kai Xiao. 2023. Implementation Analysis of Encryption and Decryption Algorithm Based on python Language. In Proceedings of the 2022 7th International Conference on Systems, Control and Communications (ICSCC '22). Association for Computing Machinery, New York, NY, USA, 1–5. https://doi.org/10.1145/3575828.3575829

Mike Hamburg, Mike Tunstall, and Qinglai Xiao. 2021. Improvements to RSA Key Generation and CRT on Embedded Devices. In Topics in Cryptology – CT-RSA 2021: Cryptographers' Track at the RSA Conference 2021, Virtual Event, May 17–20, 2021, Proceedings. Springer-Verlag, Berlin, Heidelberg, 633–656. https://doi.org/10.1007/978-3-030-75539-3_26

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = RSA algorithm
Publication Date = Last 5 years
Number of results returned =
1,710,601 Results for: All: rsa algorithm

Cristian Lupu, Bogdan Firtat, and Claudia Enoiu. 2005. Cryptography methods using the RSA algorithm. In Proceedings of the 6th WSEAS international conference on Automation & information, and 6th WSEAS international conference on mathematics and computers in biology and chemistry, and 6th WSEAS international conference on acoustics and music: theory and applications, and 6th WSEAS international conference on Mathematics and computers in business and economics (ICAI'05/MCBC'05/AMTA'05/MCBE'05). World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 271–274.

Xuewen Tan and Yunfei Li. 2012. Parallel Analysis of an Improved RSA Algorithm. In Proceedings of the 2012 International Conference on Computer Science and Electronics Engineering - Volume 01 (ICCSEE '12). IEEE Computer Society, USA, 318–320. https://doi.org/10.1109/ICCSEE.2012.286

Sushanta Kumar Sahu and Manoranjan Pradhan. 2011. Implementation of Modular Multiplication for RSA Algorithm. In Proceedings of the 2011 International Conference on Communication Systems and Network Technologies (CSNT '11). IEEE Computer Society, USA, 112–114. https://doi.org/10.1109/CSNT.2011.30

Advanced Search:
The ACM Full-Text collection
Search Within = Anywhere
Search term = integer factorization
Publication Date = Last 5 years
Number of results returned =
110,583 Results for: [All: integer factorization] AND [E-Publication Date: Past 5 years]

Adrien Poteaux and Martin Weimann. 2022. Local Polynomial Factorisation: Improving the Montes Algorithm. In Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation (ISSAC '22). Association for Computing Machinery, New York, NY, USA, 149–157. https://doi.org/10.1145/3476446.3535487

Xingbo Wang. 2022. Progress in Applying Valuated Binary Tree to Factorize Big Integers. In Proceedings of the 2022 7th International Conference on Intelligent Information Technology (ICIIT '22). Association for Computing Machinery, New York, NY, USA, 90–94. https://doi.org/10.1145/3524889.3524904

Maria Sabani, Ilias Galanis, Ilias Savvas, and Georgia Garani. 2022. Implementation of Shor's Algorithm and Reliability of Quantum Computing Devices. In Proceedings of the 25th Pan-Hellenic Conference on Informatics (PCI '21). Association for Computing Machinery, New York, NY, USA, 392–396. https://doi.org/10.1145/3503823.3503895

Daniel Chicayban Bastos and Luis Antonio Kowada. 2023. A Quantum Version of Pollard’s Rho of Which Shor’s Algorithm is a Particular Case. In Computing and Combinatorics: 28th International Conference, COCOON 2022, Shenzhen, China, October 22–24, 2022, Proceedings. Springer-Verlag, Berlin, Heidelberg, 212–219. https://doi.org/10.1007/978-3-031-22105-7_19

Factorisation (spelled with s)

*336,159 Results for: [All: semiprime factor**is**ation] AND [E-Publication Date: Past 5 years]* …….

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = RSA AND banking
Publication Date = Last 5 years
Number of results returned =
46,895 Results for: [All: rsa banking] AND [E-Publication Date: Past 5 years]

Umaprasada Rao Bodasingi and Siva Gunupuru. 2023. New Digital Signature Scheme Based on RSA Using Circulant Matrix. SN Comput. Sci. 4, 3 (Mar 2023). https://doi.org/10.1007/s42979-023-01694-4

Belbergui Chaimaa, Elkamoun Najib, and Hilal Rachid. 2021. E-banking Overview: Concepts, Challenges and Solutions. Wirel. Pers. Commun. 117, 2 (Mar 2021), 1059–1078. https://doi.org/10.1007/s11277-020-07911-0

Matus Nemec, Marek Sys, Petr Svenda, Dusan Klinec, and Vashek Matyas. 2017. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS '17). Association for Computing Machinery, New York, NY, USA, 1631–1648. https://doi.org/10.1145/3133956.3133969

Han-Byeol Park, Bo-Yeon Sim, and Dong-Guk Han. 2021. Machine Learning-Based Profiling Attack Method in RSA Prime Multiplication. In Proceedings of the 2020 ACM International Conference on Intelligent Computing and its Emerging Applications (ACM

ICEA '20). Association for Computing Machinery, New York, NY, USA, Article 32, 1–6. https://doi.org/10.1145/3440943.3444730

Search items from = The ACM Full-text collection
Search Within = Anywhere
Search term = General Number Field Sieve
Publication Date = All dates
Number of results returned = 661,837 Results for: All: general number field sieve
Hea Joung Kim and William H. Mangione-Smith. 2000. Factoring large numbers with programmable hardware. In Proceedings of the 2000 ACM/SIGDA eighth international symposium on Field programmable gate arrays (FPGA '00). Association for Computing Machinery, New York, NY, USA, 41–48. https://doi.org/10.1145/329166.329177

Search items from = The ACM Guide to Computing Literature
Search Within = Anywhere
Search term = "General Number Field Sieve" AND RSA
Publication Date = All dates
Number of results returned = 91 Results for: [All: "general number field sieve"] AND [All: rsa]

G. Harish, G. Punith Kumar, Anjan K. Koundinya, Y. V. Pramod, N. K. Srinath, G. E. Raghavendra Kumar, R. Sandeep, Archit Shukla, and Madan Acharya. 2012. Parallelization of Pollard's Rho Integer factorization algorithm. In Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology (CCSEIT '12). Association for Computing Machinery, New York, NY, USA, 43–46. https://doi.org/10.1145/2393216.2393224

Jörg Rothe. 2002. Some facets of complexity theory and cryptography: A five-lecture tutorial. ACM Comput. Surv. 34, 4 (December 2002), 504–549. https://doi.org/10.1145/592642.592646

IEEE XPLORE
Search term: Factorisation of RSA
160 results

N. Lal, A. P. Singh and S. Kumar, "Modified trial division algorithm using KNJ-factorization method to factorize RSA public key encryption," 2014 International Conference on Contemporary Computing and Informatics (IC3I), Mysore, India, 2014, pp. 992-995, doi: 10.1109/IC3I.2014.7019588.

K. Somsuk, "A new modified integer factorization algorithm using integer modulo 20's technique," 2014 International Computer Science and Engineering Conference (ICSEC), Khon Kaen, Thailand, 2014, pp. 312-316, doi: 10.1109/ICSEC.2014.6978214

B. R. Ambedkar, A. Gupta, P. Gautam and S. S. Bedi, "An Efficient Method to Factorize the RSA Public Key Encryption," *2011 International Conference on Communication Systems and Network Technologies*, Katra, India, 2011, pp. 108-111, doi: 10.1109/CSNT.2011.29.

C. Patsakis and E. Fountas, "Creating RSA Trapdoors Using Lagrange Four Square Theorem," 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kyoto, Japan, 2009, pp. 779-782, doi: 10.1109/IIH-MSP.2009.235.

K. Somsuk and S. Kasemvilas, "MVFactor: A method to decrease processing time for factorization algorithm," *2013 International Computer Science and Engineering Conference (ICSEC)*, Nakhonpathom, Thailand, 2013, pp. 339-342, doi: 10.1109/ICSEC.2013.6694805.

A. Brisson, "Rapid factorization of composite primes: An alternative to the sieve method," *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, San Francisco, CA, USA, 2017, pp. 1-3, doi: 10.1109/UIC-ATC.2017.8397602.

L. K. Galla, V. S. Koganti and N. Nuthalapati, "Implementation of RSA," *2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kumaracoil, India, 2016, pp. 81-87, doi: 10.1109/ICCICCT.2016.7987922.

<u>Search term: Factorisation of RSA + journals</u>

16 results

A. Gambhir, Simran and V. Tyagi, "Elevating ‘e’ in RSA: A Path to Improved Encryption Algorithms," *2023 3rd International Conference on Technological Advancements in Computational Sciences (ICTACS)*, Tashkent, Uzbekistan, 2023, pp. 71-75, doi: 10.1109/ICTACS59847.2023.10390067.

Google Scholar

Search: rsa algorithm and python

Denny, T., Dodson, B., Lenstra, A.K. and Manasse, M.S., 1993, August. On the factorization of RSA-120. In *Annual International Cryptology Conference* (pp. 166-174). Berlin, Heidelberg: Springer Berlin Heidelberg.

Cavallar, S., Dodson, B., Lenstra, A., Leyland, P., Lioen, W., Montgomery, P.L., Murphy, B., Te Riele, H. and Zimmermann, P., 1999. Factorization of RSA-140 using the number field sieve. In *Advances in Cryptology-ASIACRYPT’99: International Conference on the Theory and Application of Cryptology and Information Security, Singapore, November 14-18, 1999. Proceedings* (pp. 195-207). Springer Berlin Heidelberg.

Contini, S., 1999. The factorization of rsa-140. *RSA Laboratories' Bulletin*, *10*, pp.1-2.

Ambedkar, B.R., Gupta, A., Gautam, P. and Bedi, S.S., 2011, June. An efficient method to factorize the RSA public key encryption. In *2011 International Conference on Communication Systems and Network Technologies* (pp. 108-111). IEEE.

Cavallar, S. *et al.* (two thousand). Factorization of a 512-Bit RSA Modulus. In: Preneel, B. (eds) Advances in Cryptology — EUROCRYPT 2000. EUROCRYPT 2000. Lecture Notes in Computer Science, vol 1807. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45539-6_1

Boudot, F., Gaudry, P., Guillevic, A., Heninger, N., Thomé, E., Zimmermann, P. (2020). Comparing the Difficulty of Factorization and Discrete Logarithm: A 240-Digit Experiment. In: Micciancio, D., Ristenpart, T. (eds) Advances in Cryptology – CRYPTO 2020. CRYPTO 2020. Lecture Notes in Computer Science(), vol 12171. Springer, Cham. https://doi.org/10.1007/978-3-030-56880-1_3

Yeh, Y.S., Huang, T.Y., Lin, H.Y. and Chang, Y.H., 2009. A Study on Parallel RSA Factorization. *J. Comput.*, *4*(2), pp.112-118.

Seker, S.E. and Mert, C., 2013. Reverse Factorization and Comparison of Factorization Algorithms in attack to RSA. In *Proceedings of the International Conference on Scientific Computing (CSC)* (p. 173). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Lal, N., Singh, A.P. and Kumar, S., 2014, November. Modified trial division algorithm using KNJ-factorization method to factorize RSA public key encryption. In *2014 International Conference on Contemporary Computing and Informatics (IC3I)* (pp. 992-995). IEEE.

Sivakumar, J. and Begum, H., 2017. Integer factorization in RSA encryption: Challenge for cloud attackers. *International Journal of Computer Science Trends and Technology*, *5*(2), pp.405-408.

Shatnawi, A.S., Almazari, M.M., AlShara, Z., Taqieddin, E. and Mustafa, D., 2023. RSA cryptanalysis—Fermat factorization exact bound and the role of integer sequences in factorization problem. *Journal of Information Security and Applications*, *78*, p.103614.

Kleinjung, T., Aoki, K., Franke, J., Lenstra, A.K., Thomé, E., Bos, J.W., Gaudry, P., Kruppa, A., Montgomery, P.L., Osvik, D.A. and Te Riele, H., 2010. Factorization of a 768-bit RSA modulus. In *Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30* (pp. 333-350). Springer Berlin Heidelberg.

Zhou, X. and Tang, X., 2011, August. Research and implementation of RSA algorithm for encryption and decryption. In *Proceedings of 2011 6th international forum on strategic technology* (Vol. 2, pp. 1118-1121). IEEE.

Segar, T.C. and Vijayaragavan, R., 2013, July. Pell's RSA key generation and its security analysis. In *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)* (pp. 1-5). IEEE.

Karakra, A. and Alsadeh, A., 2016, July. A-rsa: augmented rsa. In *2016 SAI Computing Conference (SAI)* (pp. 1016-1023). IEEE.

Gupta, S. and Paul, G., 2009. Revisiting Fermat's Factorization for the RSA Modulus. *arXiv preprint arXiv:0910.4179*.

Balasubramanian, K., 2014, December. Variants of RSA and their cryptanalysis. In *2014 International Conference on Communication and Network Technologies* (pp. 145-149). IEEE.

(On using Euler's Factorization Algorithm to Factor RSA Modulus)
M A Budiman[1], M Zarlis[1], O S Sitompul[1] and H Mawengkang[1]
Published under licence by IOP Publishing Ltd
Journal of Physics: Conference Series, Volume 1566, 4th International Conference on Computing and Applied Informatics 2019 (ICCAI 2019) 26-27 November 2019, Medan, Indonesia Citation M A Budiman *et al* 2020 *J. Phys.: Conf. Ser.* 1566 012063
DOI 10.1088/1742-6596/1566/1/012063

Zhang, X., Li, M., Jiang, Y. and Sun, Y., 2019. A Review of the Factorization Problem of Large Integers. In *Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, July 26–28, 2019, Proceedings, Part IV 5* (pp. 202-213). Springer International Publishing.