

RSA demo

Mike May, S.J., maymk@slu.edu

A fast demo of RSA cryptography: (This is an exercise in using embedded componenets. In this example, all the code is in the action section of the buttons.)

Set a size for the primes used in terms of bits. (The size must be between 5 and 600 bits.)

Size of p1 in bits. Size of p2
 in bits.

choose primes

p1 =

```
905209907141431766616173500717327233284258723167294521115220
730763553402367610825292328936265268177319430867130559517865
171906611835843004122014027129
```

p2 =

```
588260011116558798010968248503363657205182381854042418272734
529095083260728804902547791759633834987938345845299735323894
1363334407207189738211908448809571
```

We also want to compute $n = p1 * p2$ and $\phi(n) = (p1-1) * (p2-1)$. If we let Maple choose the primes, that was done by the last button. If we insert our own primes, the next button computes n and $\phi(n)$.

compute n and phi(n)

n =

```
532498790037837808309405126234915100833860349553456382347104
288908687876992896121961517588072561515577163220284397215383
131652553288367497287075362743618298815193518313565737443436
```

phi(n) =

```
532498790037837808309405126234915100833860349553456382347104
288908687876992896121961517588072561515577163220284397215383
131652553288367497287075362743029948283086245372378107577583
```

Set the encryption key. The default is $e = 2^{16} + 1 = 65537$. Recall that $\phi(n)$ and e should be relatively prime.

e =

We now want to compute d , the inverse of $e \bmod \phi(n)$.

Compute d

$d =$

```
12036616072784120866526584280702553219941112986991932569525615966390.
45843036357968093911723593939447861960511388506816456237566376405405
76804940515494512941406054067654462150004993242960651932575923473388
65913491832755644802506278553910084176287754645724905127250559534465
2593
```

We would like to publish n and e , destroy p_1 and p_2 , and keep d a secret.

We are ready to send numerical message, Recall that the message should be less than n .

message =

```
1309110500130125
```

The ciphertext is $\text{message}^e \bmod n$.

Encrypt message

ciphertext =

```
39256487029494636893118990973681627629654897807612044597729422521978.
30590153195880454313797166061028881500419835793943469051873978385679
93153897099507642715239043553528382909667997472494041328403463808079
97415730108123935571787055434691306991369099668514131410211782399021
2465
```

We can also check deciphering

copy ciphertext for deciphering

deciphermessage=

58916951270284380355826499473259191877

The decrypted message is $\text{ciphertext}^d \bmod n$.

decode cipher message

decoded message =

1309110500130125

RSA with an ASCII message

We need to set the modulus n and the exponent, which is either d or e . We can either enter our own values or copy the values from above.

modulus =

5958676220878820368373500431851105546409952441040760951302486536109183

copy n into modulus

exponent =

2594243078113666807617087871309452592492703418689878731534648572720385

copy d into exponent

copy e into exponent

text =

This is a test of RSA

Convert text to number

number =

123362224183149454759386460341821213484243810931521

raise number to exponent

copy d into exponent

copy e into exponent

number2 =

3966930540373620721724
9803457615104119390213
0470222276386356163825

raise number2 to exponent

copy d into exponent

copy e into exponent

number3 =

1233622241831494547593
8646034182121348424381
0931521

Convert number2 to ASCII

Copy number2 back to number

Convert number3 to ASCII

Copy number3 back to number

text2 =

This is a test of RSA

Button