Simple RSA Encryption

RSA (Rivest–Shamir–Adleman) by example

Mehmet Ali Erturk - https://mehmetalierturk.com

Introduction

RSA is a widely used public-key algorithm that relies on a pair of keys — a public key (e) for encryption and a private key (d) for decryption. More information can be found on Wikipedia and Understanding Cryptography Textbook. For testing purposes and to explore how Mathematica handles RSA encryption, I would like to write a small script that encrypts and decrypts data using RSA with small numbers.

1. Generate large prime number **p** and **q**, Mathematica has built in function RandomPrime that generate prime number up to nth value.

In[ ]:=
```
maxLen = 100
```

Out[ ]=
```
100
```

```
p = RandomPrime[maxLen]
```

Out[ ]=
```
2
```

```
q = RandomPrime[maxLen]
```

Out[ ]=
```
79
```

2. Compute n = p * q

In[ ]:=
```
n = p * q
```

Out[ ]=
```
158
```

3. Compute $\phi(n) = (p-1) * (q-1)$

In[ ]:=
```
qn = (p - 1) * (q - 1)
```

Out[ ]=
```
78
```

4. Choose a public key (e), $e \in = \{1, 2, ..., \phi(n)-1\}$
such that $gcd(e, \phi(n)) = 1$

In[ ]:=
```
e = RandomPrime[qn-1]
```
Out[ ]=
```
41
```

In[ ]:=
```
GCD[e, qn]
```

5. Choose private key such that $d * e \equiv 1 \ mod \ \phi(n)$, Mathematica has built in function for modular inverses ModularInverse

In[ ]:=
```
d = ModularInverse[e,qn]
```
Out[ ]=
```
59
```

In[ ]:=
```
Mod[e * d, qn]
```
Out[ ]=
```
1
```

6. Given with private key **d**, and **n**, decryption of cipher y, is $x = y^{d} \ mod \ n$

In[ ]:=
```
x' = Mod [Power[y, d], n ]
```
Out[ ]=
```
7
```

In[ ]:=
```
x == x'
```
Out[ ]=
```
True
```