Alice and Bob’s Adventures in Cryptoland...
Information

Theoretical

Cryptography
Information Theoretical Cryptography

Key Distribution

Encryption
Will you marry me?
Will you marry me?

Divorce your wife first!
Will you marry me?

Divorce your wife first!

The papers are in the mail...
Will you marry me?

Divorce your wife first!

The papers are in the mail...

OK, I will!
Key Distribution
Encryption
Will you marry me?

Decryption

Encryption

8RdewtU5qkLa$es!T9@

8RdewtU5qkLa$ by me?
Will you marry me?

8RdewtU5qkLa$es!T9@

Decryption

Encryption

8RdewtU5qkLa$es!T9@

Will you marry me?
Encrypt

Decrypt

Do you really want to encrypt!

8RdewtU5qkLa$es!T9@

l(D%eXhDqliykl#2cV7dEwnMs
Encrpytion 8RdewtU5qkLa$es!T9@

Decrpytion l(D%eXhDqliykl#2cV7dEwnMs

Divorce your wife first!
Symmetric Encryption

Encryption

Decryption

Information Theoretical Security
Symmetric Encryption

Ceasar’s Cipher
VERNAM’s Cipher

1910’s
m ⊕ k

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
VERNAM’s Cipher

\[ m \oplus k = c \]

\[
\begin{array}{cccc}
1 & 1 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 1 & 1 \\
0 & 1 & 1 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
\end{array}
\]
**VERNAM’s Cipher**

\[ m \oplus k = c \]

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

\[ \oplus = \]

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
VERNAM’s Cipher

\[ m \oplus k = c \]

\[
\begin{array}{ccc}
1 & 1 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
\end{array}
\]

\[ c \oplus k \]

\[
\begin{array}{ccc}
0 & 1 & 0 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
0 & 1 & 1 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 1 \\
1 & 0 & 1 \\
\end{array}
\]
VERNAM's Cipher

\[ m \oplus k = c \]

\[
\begin{array}{cccc}
1 & 1 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
0 & 0 & 0 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0
\end{array}
\]

\[ c \oplus k = m \]

\[
\begin{array}{cccc}
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
0 & 1 & 1 \\
0 & 1 & 1
\end{array}
\]
VERNAM’s One-Time Pad

\[ m_1 \oplus k = c_1 \]
\[ m_2 \oplus k = c_2 \]

\[ c_1 \oplus c_2 = m_1 \oplus m_2 \]
Complexity

Theoretical

Cryptography
The Enigma Machine

Arthur Scherbius

1930’s
Asymmetric Encryption
(Public-Key Cryptography)

Encryption

Decryption

P

K_e

K_d

C

Complexity Theoretical Security
Public-Key Cryptography

8RdewtU5qkLa$es!T9@
Public-Key Cryptography

Encryption

8RdewtU5qkLa$es!T9@

8RdewtU5qkLa$es!T9@

Encrypyytttiioonn

PPuubblliicc--KKeeyyCCrryyppttooggrraapphhyy

Decryption

WWiilllyyoouummaarrrryymmee??
Public-Key Cryptography

Will you marry me? Yes! T9@

8RdewtU5qkLa$es!T9@

Decryption

Encryption
Fast Exponentiation

Input: base $x$ and exponent $e$.

Output: $x^e$. // *** warning $|x^e| \sim |e||x|$ ***

$y = 1$
WHILE $e > 0$ DO
  IF $e \% 2 = 1$ THEN $y = xy$
  $e = e/2; x = x^2$
return $y$

running time is $O(e|x|^2) = O(2^{|e|}|x|^2)$
Fast Modular Exponentiation

- **Input:** base \( x \), modulus \( N \) and exponent \( e \).
- **Output:** \( x^e \mod N \).

\[
y = 1
\]

```plaintext
WHILE \( e > 0 \) DO
  IF \( e \% 2 = 1 \) THEN
    \( y = xy \mod N \)
  \( e = e/2; x = x^2 \mod N \)

return \( y \)
```

- Running time is \( O(|e|*|x|^2) = O(|x|^3) \)
Example

$5^{13} \mod 7$
Example

$5^{13} \mod 7 = 5^{8+4+1} \mod 7$
Example

\[ 5^{13} \mod 7 \]
\[ = 5^{8+4+1} \mod 7 \]
\[ = 5^8 \times 5^4 \times 5^1 \mod 7 \]
Example

$5^{13} \mod 7$

$= 5^{8+4+1} \mod 7$

$= 5^8 \times 5^4 \times 5^1 \mod 7$

$5^0, 5^1, 5^2, 5^4, 5^8$
Example

\[ 5^{13} \mod 7 \]

\[ = 5^{8+4+1} \mod 7 \]

\[ = 5^8 \times 5^4 \times 5^1 \mod 7 \]

\[ 5^0, 5^1, 5^2, 5^4, 5^8 \]

\[ \equiv 1, 5, 4, 2, 4 \ (\mod 7) \]
Example

$$5^{13} \mod 7$$

$$= 5^{8+4+1} \mod 7$$

$$= 5^8 \times 5^4 \times 5^1 \mod 7$$

$$5^0, 5^1, 5^2, 5^4, 5^8$$

$$\equiv 1, 5, 4, 2, 4 \pmod 7$$

$$= 4 \times 2 \times 5 \mod 7$$
Example

\[ 5^{13} \mod 7 \]

\[ = 5^{8+4+1} \mod 7 \]

\[ = 5^8 \times 5^4 \times 5^1 \mod 7 \]

\[ 5^0, 5^1, 5^2, 5^4, 5^8 \]

\[ \equiv 1, 5, 4, 2, 4 \pmod{7} \]

\[ = 4 \times 2 \times 5 \mod 7 \]

\[ = 1 \times 5 \mod 7 \]
Example

\[ 5^{13} \mod 7 \]

\[ = 5^{8+4+1} \mod 7 \]

\[ = 5^8 \times 5^4 \times 5^1 \mod 7 \]

\[ 5^0, 5^1, 5^2, 5^4, 5^8 \]

\[ \equiv 1, 5, 4, 2, 4 \pmod{7} \]

\[ = 4 \times 2 \times 5 \mod 7 \]

\[ = 1 \times 5 \mod 7 \]

\[ = 5 \]
Note. \( \gcd(a, b) = g \rightarrow \exists x, y \in \mathbb{Z} \text{ such that } g = ax + by. \) The following recursive definition is based on the property \( \gcd(a, b) = \gcd(a, b - a). \)

\[
\gcd(a, b) = \begin{cases} 
a & \text{if } b = 0 \\
gcd(b, a \mod b) & \text{otherwise}
\end{cases}
\]
Euclidian Algorithm

- **Input**: integers $a, b$.
- **Output**: $g, x, y$ such that $g = \text{GCD}(a, b)$.

\[
g = a; \quad g' = b; \\
\text{WHILE } g' > 0 \text{ DO} \\
\quad k = g / g' \\
\quad g'' = g - kg'; \\
\quad g = g'; \\
\quad g' = g''; \\
\text{return } g
\]

Running time is $O(|a| \times |b|)$
EXAMPLE
EXAMPLE

gcd(7,19)
EXAMPLE

gcd(7,19)
= gcd(19,7)
EXAMPLE

gcd(7,19)
= gcd(19,7)
= gcd(7,19 \ mod \ 7)
EXAMPLE

gcd(7,19)
= gcd(19,7)
= gcd(7,19 \ mod \ 7)
= gcd(7,5)
EXAMPLE

\[ \text{gcd}(7, 19) \]
\[ = \text{gcd}(19, 7) \]
\[ = \text{gcd}(7, 19 \text{ mod } 7) \]
\[ = \text{gcd}(7, 5) \]
\[ = \text{gcd}(5, 7 \text{ mod } 5) \]
EXAMPLE

\[ \gcd(7, 19) \]
\[ = \gcd(19, 7) \]
\[ = \gcd(7, 19 \mod 7) \]
\[ = \gcd(7, 5) \]
\[ = \gcd(5, 7 \mod 5) \]
\[ = \gcd(5, 2) \]
EXAMPLE

gcd(7, 19)
= gcd(19, 7)
= gcd(7, 19 \ mod \ 7)
= gcd(7, 5)
= gcd(5, 7 \ mod \ 5)
= gcd(5, 2)
= gcd(2, 5 \ mod \ 2)
EXAMPLE

gcd(7, 19)
= gcd(19, 7)
= gcd(7, 19 \text{ mod } 7)
= gcd(7, 5)
= gcd(5, 7 \text{ mod } 5)
= gcd(5, 2)
= gcd(2, 5 \text{ mod } 2)
= gcd(2, 1)
EXAMPLE

gcd(7, 19)
= gcd(19, 7)
= gcd(7, 19 \ mod \ 7)
= gcd(7, 5)
= gcd(5, 7 \ mod \ 5)
= gcd(5, 2)
= gcd(2, 5 \ mod \ 2)
= gcd(2, 1)
= gcd(1, 2 \ mod \ 1) = gcd(1, 0) = 1
The idea behind the following iterative algorithm is to maintain in each iteration the relations \( g = ax + by \) and \( g' = ax' + by' \) while reducing the value of \( g \).

At the end of the algorithm, the value of \( g \) is \( \gcd(a, b) \). The final value of \( x \) is such that \( ax \equiv g \pmod{b} \) and by symmetry, the final value of \( y \) is such that \( by \equiv g \pmod{a} \). When \( \gcd(a, b) = 1 \), we find that \( x \) is the multiplicative inverse of \( a \) modulo \( b \) and that \( y \) is the multiplicative inverse of \( b \) modulo \( a \).
Extended Euclidean Algorithm

Input: integers \(a, b\).

Output: \(g, x, y\) such that \(g = \text{GCD}(a, b)\) and \(g = ax + by\).

\[
g = a; g' = b; x = 1; y = 0; x' = 1; y' = 1;
\]

WHILE \(g' > 0\) DO

\[
k = g / g'
\]

\[
g'' = g - kg'; x'' = x - kx'; y'' = y - ky'; \quad // g'' = g \mod g'
\]

\[
g = g'; x = x'; y = y';
\]

\[
g' = g''; x' = x''; y' = y'';
\]

return \(g, x, y\)

Running time is \(O(|a| \times |b|)\)
EXAMPLE
EXAMPLE

$\gcd(7, 19)$

$x = 1, y = 0, x' = 0, y' = 1$
EXAMPLE

\[ \text{gcd}(7, 19) \]
\[ = \text{gcd}(19, 7) \]
\[ x = 1, y = 0, x' = 0, y' = 1 \]
\[ x = 0, y = 1, x' = 1, y' = 0 \]
EXAMPLE

gcd(7,19)  
= gcd(19,7)  
= gcd(7,19 \ mod \ 7)  

x=1, y=0, x' = 0, y' = 1  
x=0, y=1, x' = 1, y' = 0
EXAMPLE

\[
\begin{align*}
\gcd(7, 19) &= x=1, y=0, x' = 0, y' = 1 \\
= \gcd(19, 7) &= x=0, y=1, x' = 1, y' = 0 \\
= \gcd(7, 19 \mod 7) &= x=1, y=0, x' = -2, y' = 1 \\
= \gcd(7, 5)
\end{align*}
\]
EXAMPLE

gcd(7,19) = gcd(19,7) = gcd(7,19 \mod 7) = gcd(7,5) = gcd(5,7 \mod 5)

x=1, y=0, x'=0, y'=1
x=0, y=1, x'=1, y'=0
x=1, y=0, x'=-2, y'=1
EXAMPLE

gcd(7,19)        x=1, y=0, x' = 0, y' = 1
= gcd(19,7)      x=0, y=1, x' = 1, y' = 0
= gcd(7, 19 \mod 7)
= gcd(7, 5)      x=1, y=0, x' = -2, y' = 1
= gcd(5, 7 \mod 5)
= gcd(5, 2)      x=-2, y=1, x' = 3, y' = -1
EXAMPLE

\[ \gcd(7, 19) = \gcd(19, 7) = \gcd(7, 19 \mod 7) = \gcd(7, 5) = \gcd(5, 7 \mod 5) = \gcd(5, 2) = \gcd(2, 5 \mod 2) \]

\[
\begin{align*}
x &= 1, y &= 0, x' &= 0, y' &= 1 \\
x &= 0, y &= 1, x' &= 1, y' &= 0 \\
x &= 1, y &= 0, x' &= -2, y' &= 1 \\
x &= -2, y &= 1, x' &= 3, y' &= -1
\end{align*}
\]
EXAMPLE

gcd(7,19) x=1, y=0, x' = 0, y' = 1
= gcd(19, 7) x=0, y=1, x' = 1, y' = 0
= gcd(7, 19 mod 7)
= gcd(7, 5) x=1, y=0, x' = -2, y' = 1
= gcd(5, 7 mod 5)
= gcd(5, 2) x=-2, y=1, x' = 3, y' = -1
= gcd(2, 5 mod 2)
= gcd(2, 1) x=3, y=-1, x' = -8, y' = 3
EXAMPLE

\[ \text{gcd}(7, 19) \]
\[ = \text{gcd}(19, 7) \]
\[ = \text{gcd}(7, 19 \mod 7) \]
\[ = \text{gcd}(7, 5) \]
\[ = \text{gcd}(5, 7 \mod 5) \]
\[ = \text{gcd}(5, 2) \]
\[ = \text{gcd}(2, 5 \mod 2) \]
\[ = \text{gcd}(2, 1) \]
\[ = \text{gcd}(1, 2 \mod 1) \]

\[ x = 1, y = 0, x' = 0, y' = 1 \]
\[ x = 0, y = 1, x' = 1, y' = 0 \]
\[ x = 1, y = 0, x' = -2, y' = 1 \]
\[ x = -2, y = 1, x' = 3, y' = -1 \]
\[ x = 3, y = -1, x' = -8, y' = 3 \]
EXAMPLE

\[
\begin{align*}
gcd(7, 19) &= x=1, y=0, x'=0, y'=1 \\
= gcd(19, 7) &= x=0, y=1, x'=1, y'=0 \\
= gcd(7, 19 \mod 7) &= x=1, y=0, x'=-2, y'=1 \\
= gcd(7, 5) &= x=1, y=0, x'=1, y'=0 \\
= gcd(5, 7 \mod 5) &= x=1, y=0, x'=-2, y'=1 \\
= gcd(5, 2) &= x=1, y=0, x'=1, y'=0 \\
= gcd(2, 5 \mod 2) &= x=1, y=0, x'=1, y'=0 \\
= gcd(2, 1) &= x=1, y=0, x'=1, y'=0 \\
= gcd(1, 2 \mod 1) &= x=1, y=0, x'=1, y'=0 \\
= gcd(1, 0) &= x=1, y=0, x'=1, y'=0 \\
= 1 &
\end{align*}
\]
EXAMPLE

gcd(7, 19)
EXAMPLE

gcd(7,19) = gcd(1,0) = 1

x = -8, y = 3, x' = 19, y' = -7
EXAMPLE

gcd(7,19) = gcd(1,0) = 1

x = -8, y = 3, x' = 19, y' = -7
EXAMPLE

\[ \text{gcd}(7, 19) = \text{gcd}(1, 0) = 1 \]

\[ x = -8, y = 3, x' = 19, y' = -7 \]

thus
EXAMPLE

\[ \operatorname{gcd}(7,19) = \operatorname{gcd}(1,0) = 1 \]

\[ x = -8, y = 3, x' = 19, y' = -7 \]

thus

\[ 1 = -8 \times 7 + 3 \times 19 \]
EXAMPLE

\[\text{gcd}(7,19) = \text{gcd}(1,0) = 1\]

\[x = -8, y = 3, x' = 19, y' = -7\]

thus

\[1 = -8 \times 7 + 3 \times 19\]

and
EXAMPLE

gcd(7,19)
= gcd(1,0) = 1

thus

\[ 1 = -8 \times 7 + 3 \times 19 \]

and

\[ 7^{-1} \mod 19 = 11 = -8 \mod 19 \]
EXAMPLE

\[ \text{gcd}(7, 19) \]
\[ = \text{gcd}(1, 0) = 1 \]
\[ x = -8, y = 3, x' = 19, y' = -7 \]

thus

\[ 1 = -8 \times 7 + 3 \times 19 \]

and

\[ 7^{-1} \mod 19 = 11 = -8 \mod 19 \]
\[ 19^{-1} \mod 7 = 3 = 3 \mod 7 \]
**Primality Testing**

- **Input:** base $a$, modulus $N$.
- **Output:** Is $N$ a base-$a$ pseudo-prime?

IF $\gcd(a,N) > 1$ THEN return False

set $s \geq 0$ and $t$ (odd) s.t. $N-1 = t2^s$

$x = a^2 \%N; y = N-1$

FOR $i = 1$ TO $s$

IF $x = 1$ AND $y = N-1$ THEN return True

$y = x; x = x^2 \%N$

return False

running time is $O(|N|^4)$
Asymmetric Encryption
(Public-Key Cryptography)

Encryption

\[ K_e \]

Decryption

\[ K_d \]

P

C

Complexity Theoretical Security
Definitions

**DEFINITION** A public-key encryption scheme is a tuple of PPT algorithms \((\text{Gen}, \text{Enc}, \text{Dec})\) s.t.:

1. The key generation algorithm \(\text{Gen}\) takes as input the security parameter \(1^n\) and outputs a pair of keys \((pk, sk)\). We refer to the first of these as the public key and the second as the private key. We assume for convenience that \(pk\) and \(sk\) each have length at least \(n\), and that \(n\) can be determined from \(pk, sk\).
2. The encryption algorithm $\text{Enc}$ takes as input a public key $pk$ and a message $m$ from some underlying plaintext space. It outputs a ciphertext $c$, and we write this as $c \leftarrow \text{Enc}_{pk}(m)$.

3. The decryption algorithm $\text{Dec}$ takes as input a private key $sk$ and a ciphertext $c$, and outputs a message $m$ or a special symbol $\perp$ denoting failure. We assume without loss of generality that $\text{Dec}$ is deterministic, and write this as $m := \text{Dec}_{sk}(c)$. 
It is required that there exists a negligible function $\text{negl}$ such that for every $n$, every $(pk, sk)$ output by $\text{Gen}(1^n)$, and every message $m$ in the appropriate underlying plaintext space, it holds that

$$\Pr[\text{Dec}_{sk}(\text{Enc}_{pk}(m)) \neq m] \leq \text{negl}(n).$$
RSA Encryption

Private inventors
Ellis, Cocks, Williamson

Public inventors

1970’s
In Cocks’ variation, \( e = N \) and therefore \( d = N^{-1} \mod \varphi(N) \).
RSA Encryption

Gen: on input $1^n$ run GenRSA($1^n$) and obtain $(N,e,d)$. Let $\langle N,e \rangle$ be the public-key and $\langle d \rangle$ the private key.

Enc: on input $\langle N,e \rangle$ and a message $0 < m < N$ compute
\[ c = m^e \mod N \]

Dec: on input $\langle d \rangle$ and a ciphertext $0 < c < N$ compute
\[ m = c^d \mod N \]
The RSA Assumption

The RSA problem can be described informally as follows: given a modulus \( N \), an integer (exponent) \( e > 0 \) that is relatively prime to \( \varphi(N) \), and an element \( y \in \mathbb{Z}_N^* \), compute \( e \sqrt{y} \mod N \);

Given \( N, e, y \) find \( x \) such that \( x^e = y \mod N \).
The RSA Assumption

The RSA experiment $\text{RSA-inv}_{A, \text{GenRSA}(n)}$:

1. Run $\text{GenRSA}(1^n)$ to obtain $(N,e,d)$.

2. Choose $y \leftarrow \mathbb{Z}_N^*$. 

3. $A$ is given $N,e,y$, and outputs $x \in \mathbb{Z}_N^*$.

4. The output of the experiment is defined to be $1$ if $y = x^e \mod N$, and $0$ otherwise.
The RSA Assumption

**DEFINITION** We say that the RSA problem is hard relative to GenRSA if for all probabilistic polynomial-time algorithms $A$ there exists a negligible function $\text{negl}$ such that

$$\Pr[\text{RSA-inv}_{A,\text{GenRSA}(n)} = 1] \leq \text{negl}(n).$$
Quantum Factoring
Approximate Integer GCD Based Cryptography

§

2010’s
Approximate Integer GCD

0  p  2p  3p  4p  5p
Approximate Integer GCD

0  p  2p  3p  4p  5p
Approximate Integer GCD

0      p      2p    3p     4p    5p

0      p      2p    3p     4p    5p
Approximate Integer GCD

0      p      2p    3p     4p    5p
Approximate Integer GCD

0  p  2p  3p  4p  5p

0  p  2p  3p  4p  5p
Approximate Integer GCD

0      p      2p    3p     4p    5p

0      p      2p    3p     4p    5p
Approximate Integer GCD

GCD(q_1p,q_2p,q_3p,q_4p) = p
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
Approximate Integer GCD

\[
\begin{align*}
X_1 & \quad q_1p \\
X_2 & \quad q_2p \\
X_3 & \quad q_3p \\
X_4 & \quad q_4p \\
\end{align*}
\]

\[
\begin{align*}
Z_1 & \quad q_1p \pm 2e_1 \\
Z_2 & \quad q_2p \pm 2e_2 \\
Z_3 & \quad q_3p \pm 2e_3 \\
Z_4 & \quad q_4p \pm 2e_4 \\
\end{align*}
\]

\[\text{GCD}(x_1, x_2, x_3, x_4) = p\]
Approximate Integer GCD

\[ \text{GCD}(x_1, x_2, x_3, x_4) = p \]
\[ \text{GCD}(z_1, z_2, z_3, z_4) = 1 \]
Approximate Integer GCD

\[
GCD(x_1, x_2, x_3, x_4) = p
\]
Approximate Integer GCD

GCD\((x_1,x_2,x_3,x_4) = p\)

\text{AIGCD} : \text{find } p \text{ from } z_1, z_2, z_3, z_4 \ ?
Approximate Integer GCD

$Z_1 \quad Z_2 \quad Z_3 \quad \ldots \quad Z_{k-1} \quad Z_k \quad Z_0$
Approximate Integer GCD

\[ \sum_{1 \leq i \leq k} s_i z_i \mod z_0 \]
Approximate Integer GCD

\[ \sum_{1 \leq i \leq k} s_i z_i \mod z_0 \]

\[ s_i \in \{0, 1\} \]
Approximate Integer GCD

\[ \sum_{1 \leq i \leq k} s_i z_i \text{ mod } z_0 \]

\[ s_i \in \{0, 1\} \]
Approximate Integer GCD

\[
\sum_{1 \leq i \leq k} s_i z_i \mod z_0
\]

\( s_i \in \{0, 1\} \)
Approximate Integer GCD

\[ \sum_{i \leq k} s_i z_i \mod z_0 \approx \sum_{i \leq k} s_i x_i \mod x_0 \]

\[ s_i \in \{0, 1\} \]
Approximate Integer GCD

\[ \sum_{i=1}^{k} s_i z_i \mod z_0 \approx \sum_{i=1}^{k} s_i x_i \mod x_0 \]

\[ s_i \in \{0, 1\} \quad \pm 2(ke_0 + \sum_{i=1}^{k} e_i) \]
Approximate Integer GCD

\[
\left| \sum_{1 \leq i \leq k} s_i z_i \mod z_0 - \left( \sum_{1 \leq i \leq k} s_i q_i \mod q_0 \right) \times p \right| \leq 4k|e_{\text{max}}|
\]

\[s_i \in \{0, 1\}\]
Approximate Integer GCD
Approximate Integer GCD
Approximate Integer GCD

$$\Omega(s) = \sum_{1 \leq i \leq k} s_i z_i \mod z_0$$
Approximate Integer GCD

\[ \Omega(s) = \sum_{1 \leq i \leq k} s_i z_i \mod z_0 \]

\[ s \in \{0,1\}^n \]
Approximate Integer GCD

\[ \Omega(s) = \sum_{1 \leq i \leq k} s_i z_i \mod z_0 \]

\[ s \in \{0,1\}^n \]

\[ |e_{\max}| \leq \partial \ll p/8k \]
Approximate Integer GCD

$$\Omega(s) = \sum_{1 \leq i \leq k} s_i z_i \mod z_0$$

$$s \in \{0, 1\}^n$$

$$|e_{\text{max}}| \leq \partial \ll p/8k$$

$$\Omega(s) - p[\Omega(s)/p] = \text{small even error}$$
AIGCD encryption
AIGCD encryption

SK : p
AIGCD encryption

SK : p
PK : z₀, z₁, z₂,..., zₖ, \( \partial \ll p/8k \ll \partial' \ll p/2 \)
AIGCD encryption

SK : $p$
PK : $z_0, z_1, z_2, \ldots, z_k, \vartheta \ll p/8k \ll \vartheta' \ll p/2$

$e_i \in \mathbb{U} [-\vartheta \ldots + \vartheta]$
AIGCD encryption

SK : p
PK : z₀, z₁, z₂, ..., zₖ, \( \partial \ll p/8k \ll \partial' \ll p/2 \)
\( e_i \in_U [-\partial'...+\partial] \)

\[ enc(b) = \Omega(s)+2e+b \]
\( s \in_U \{0,1\}^n \)
\( e \in_U [-\partial'...+\partial'] \)
AIGCD encryption

SK : p
PK : z₀, z₁, z₂,..., zₖ, \( \partial \ll p/8k \ll \partial' \ll p/2 \)

\[ e_i \in_U [-\partial'...+\partial] \]

\[ \text{enc}(b) = \Omega(s)+2e+b \]
\[ s \in_U \{0,1\}^n \]
\[ e \in_U [-\partial'...+\partial'] \]

\[ \text{dec}(c) = c-p[c/p] \mod 2 \]
\[ = \text{parity of error} \]
Cryptography Today

In the current global environment, rapid and secure information sharing is important to protect our Nation, its citizens and its interests. Strong cryptographic algorithms and secure protocol standards are vital tools that contribute to our national security and help address the ubiquitous need for secure, interoperable communications.

Currently, Suite B cryptographic algorithms are specified by the National Institute of Standards and Technology (NIST) and are used by NSA's Information Assurance Directorate in solutions approved for protecting classified and unclassified National Security Systems (NSS). Below, we announce preliminary plans for transitioning to quantum resistant algorithms.
Updates

February 24, 2016: The slides of NIST Announcement
"Post-Quantum Cryptography: NIST’s Plan for the Future"
by Dustin Moody are now available.

February 21, 2016: Useful information concerning the winter school and the conference can be found in the public webfolder: https://goo.gl/UGwjua (access is read-only)

February 15, 2016: Due to high attendance, for participants, who register on site, only access to an electronic version (instead of a hardcopy) of the proceedings will be provided.

February 10, 2016: Updated programs of the conference (with hot topic session included) and the winter school are now available.

February 9, 2016: Deadline for online registration has passed.
On-site registration will be available at the venue from February 22.
Post–Quantum Cryptography: NIST’s Plan for the Future

Dustin Moody
Post Quantum Cryptography Team
National Institute of Standards and Technology (NIST)

Timeline

- Fall 2016 – formal Call For Proposals
- Nov 2017 – Deadline for submissions
- 3–5 years – Analysis phase
  - NIST will report its findings
- 2 years later – Draft standards ready

- Workshops
  - Early 2018 – submitter’s presentations
  - One or two during the analysis phase