CS70: Lecture 9. Outline.

1. Public Key Cryptography
2. RSA system
   2.1 Efficiency: Repeated Squaring.
   2.2 Correctness: Fermat's Theorem.
   2.3 Construction.
3. Warnings.

Isomorphisms.

Bijection:

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where

$x = a \pmod{m}$ and $x = b \pmod{n}$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where

$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a',b') = (2,4)$, then $x = 22 \pmod{45}$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a',b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider:

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a',b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65$

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$?

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes!

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes! Is it $2 \pmod{9}$?

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a',b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes! Is it $2 \pmod{9}$? Yes!

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes! Is it $2 \pmod{9}$? Yes!

Isomorphism:

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:

If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where

$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a', b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes! Is it $2 \pmod{9}$? Yes!

Isomorphism:

the actions under $\pmod{5}$, $\pmod{9}$

Isomorphisms.

Bijection:

$f(x) = ax \pmod{m}$ if $gcd(a,m) = 1$.

Simplified Chinese Remainder Theorem:
If $gcd(n,m) = 1$, there is unique $x \pmod{mn}$ where
$x = a \pmod{m}$ and $x = b \pmod{n}$.

Bijection between $(a \pmod{n}, b \pmod{m})$ and $x \pmod{mn}$.

Consider $m = 5$, $n = 9$, then if $(a,b) = (3,7)$ then $x = 43 \pmod{45}$.

Consider $(a',b') = (2,4)$, then $x = 22 \pmod{45}$.

Now consider: $(a,b) + (a',b') = (0,2)$.

What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

Try $43 + 22 = 65 = 20 \pmod{45}$.

Is it $0 \pmod{5}$? Yes! Is it $2 \pmod{9}$? Yes!

Isomorphism:
the actions under $\pmod{5}$, $\pmod{9}$
correspond to actions in $\pmod{45}$!

Poll

$x = 5 \mod 7$ **and** $x = 5 \mod 6$
$y = 4 \mod 7$ **and** $y = 3 \mod 6$

Poll

$x = 5 \mod 7$ **and** $x = 5 \mod 6$
$y = 4 \mod 7$ **and** $y = 3 \mod 6$

What's true?

Poll

$x = 5 \pmod 7$ **and** $x = 5 \pmod 6$
$y = 4 \pmod 7$ **and** $y = 3 \pmod 6$

What's true?

(A) $x + y = 2 \pmod 7$
(B) $x + y = 2 \pmod 6$
(C) $xy = 3 \pmod 6$
(D) $xy = 6 \pmod 7$
(E) $x = 5 \pmod{42}$
(F) $y = 39 \pmod{42}$

Poll

$x = 5 \mod 7$ **and** $x = 5 \mod 6$
$y = 4 \mod 7$ **and** $y = 3 \mod 6$

What's true?

(A) $x + y = 2 \mod 7$
(B) $x + y = 2 \mod 6$
(C) $xy = 3 \mod 6$
(D) $xy = 6 \mod 7$
(E) $x = 5 \mod 42$
(F) $y = 39 \mod 42$

All true.

Xor

Computer Science:

Xor

Computer Science:

1 - True

0 - False

Xor

Computer Science:

1 - True
0 - False

$1 \vee 1 = 1$

Xor

Computer Science:

1 - True

0 - False

$1 \vee 1 = 1$

$1 \vee 0 = 1$

$0 \vee 1 = 1$

$0 \vee 0 = 0$

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!

Xor

Computer Science:

1 - True

0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!

$\{0, 1\}$ is set. Take remainder for 2.

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!
$\{0, 1\}$ is set. Take remainder for 2.

Xor

Computer Science:

1 - True

0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!

$\{0, 1\}$ is set. Take remainder for 2.

Property: $A \oplus B \oplus B = A$.

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!
$\{0, 1\}$ is set. Take remainder for 2.

Property: $A \oplus B \oplus B = A$.
By cases: $1 \oplus 1 \oplus 1 = 1$.

Xor

Computer Science:
 1 - True
 0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

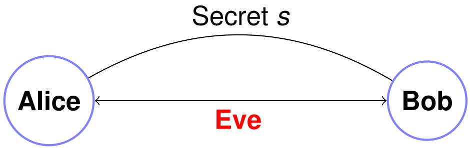
$A \oplus B$ - Exclusive or.
$1 \oplus 1 = 0$
$1 \oplus 0 = 1$
$0 \oplus 1 = 1$
$0 \oplus 0 = 0$

Note: Also modular addition modulo 2!
 $\{0, 1\}$ is set. Take remainder for 2.

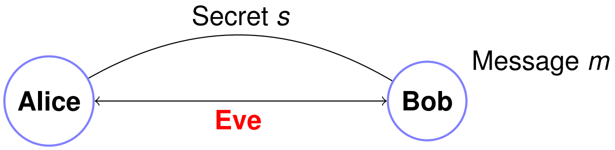
Property: $A \oplus B \oplus B = A$.
By cases: $1 \oplus 1 \oplus 1 = 1$. $\ldots$

Cryptography ...

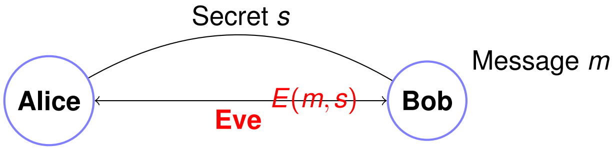
Cryptography ...



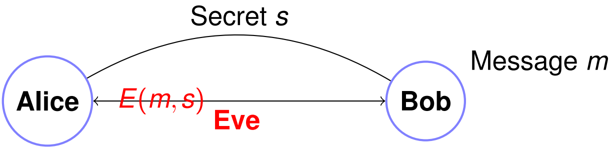
Cryptography ...



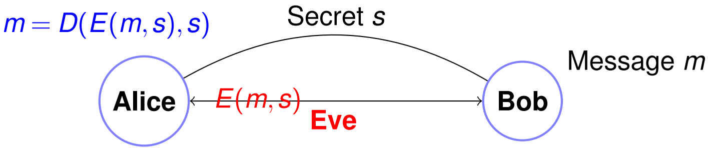
Cryptography ...



Cryptography ...

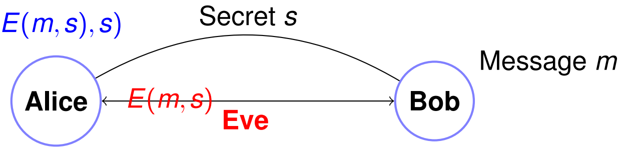


Cryptography ...



Cryptography ...

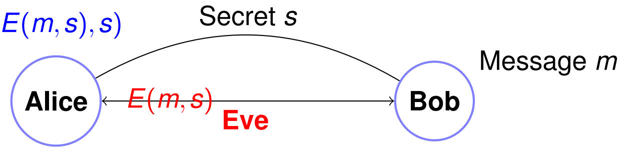
$m = D(E(m,s),s)$



Example:

Cryptography ...

$m = D(E(m,s),s)$

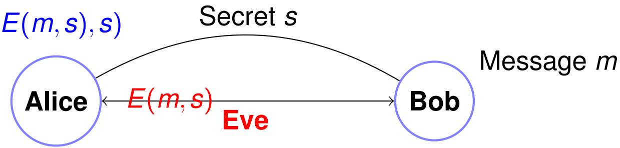


Example:

One-time Pad: secret s is string of length $|m|$.

Cryptography ...

$m = D(E(m,s),s)$



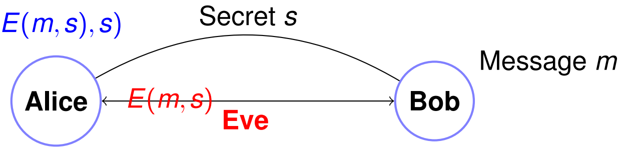
Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

Cryptography ...

$m = D(E(m,s),s)$



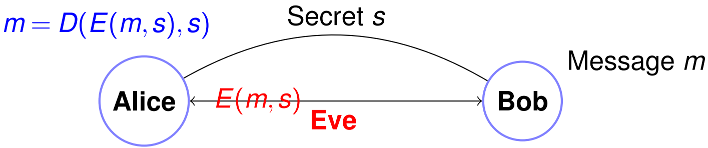
Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

$s = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

Cryptography ...



Example:

One-time Pad: secret s is string of length $|m|$.

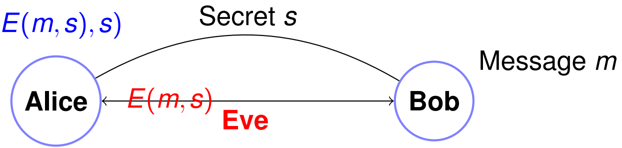
$m = 1010101110101101$

$s\ = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$E(m,s)$ – bitwise $m \oplus s$.

Cryptography ...

$m = D(E(m,s),s)$



Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

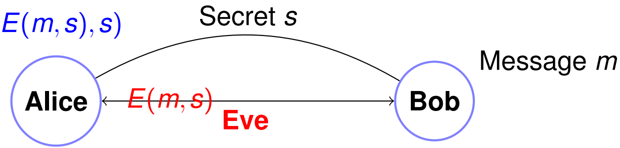
$s\ = ...$

$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

Cryptography ...

$m = D(E(m,s),s)$



Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

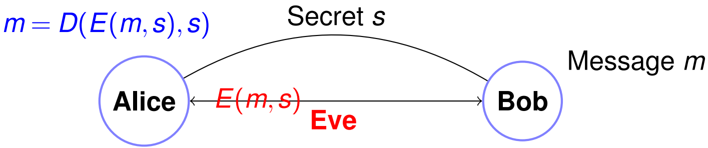
$s\ = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

Works because $m \oplus s \oplus s = m$!

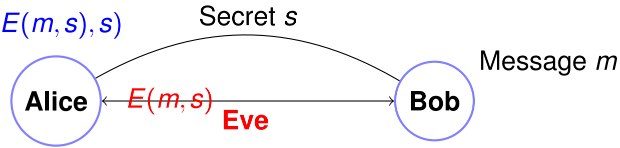
Cryptography ...



Example:
One-time Pad: secret s is string of length $|m|$.
$m = 1010101110101101$
$s\ = \ldots\ldots\ldots\ldots\ldots\ldots$
$E(m,s)$ – bitwise $m \oplus s$.
$D(x,s)$ – bitwise $x \oplus s$.
Works because $m \oplus s \oplus s = m$!
...and totally secure!

Cryptography ...

$m = D(E(m,s),s)$



Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

s = ...

$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

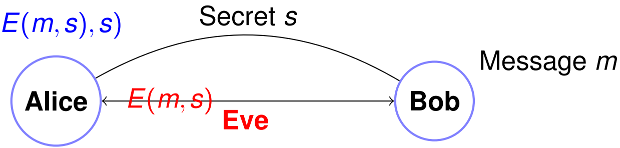
Works because $m \oplus s \oplus s = m$!

...and totally secure!

...given $E(m,s)$ any message m is equally likely.

Cryptography ...

$m = D(E(m,s),s)$



Example:
One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

$s = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

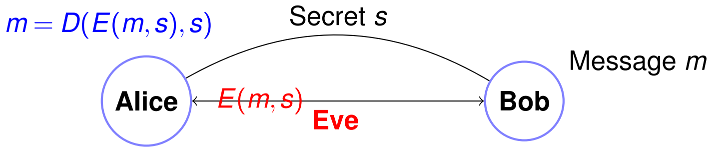
$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

Works because $m \oplus s \oplus s = m$!
...and totally secure!
...given $E(m,s)$ any message m is equally likely.

Disadvantages:

Cryptography ...



Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

$s = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

Works because $m \oplus s \oplus s = m$!

...and totally secure!

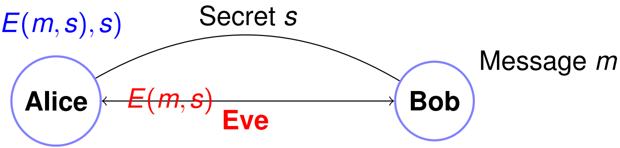
...given $E(m,s)$ any message m is equally likely.

Disadvantages:

Shared secret!

Cryptography ...

$m = D(E(m,s),s)$



Example:

One-time Pad: secret s is string of length $|m|$.

$m = 1010101110101101$

$s = \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$E(m,s)$ – bitwise $m \oplus s$.

$D(x,s)$ – bitwise $x \oplus s$.

Works because $m \oplus s \oplus s = m$!

...and totally secure!

...given $E(m,s)$ any message m is equally likely.

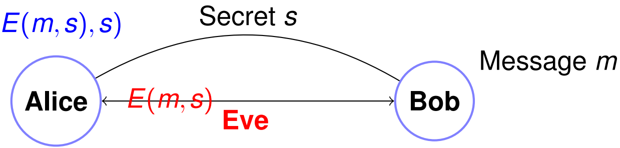
Disadvantages:

Shared secret!

Uses up one time pad..

Cryptography ...

$m = D(E(m,s),s)$



Example:
One-time Pad: secret s is string of length $|m|$.
$m = 1010101110101101$
$s = \ldots\ldots\ldots\ldots\ldots\ldots$
$E(m,s)$ – bitwise $m \oplus s$.
$D(x,s)$ – bitwise $x \oplus s$.
Works because $m \oplus s \oplus s = m$!
...and totally secure!
...given $E(m,s)$ any message m is equally likely.

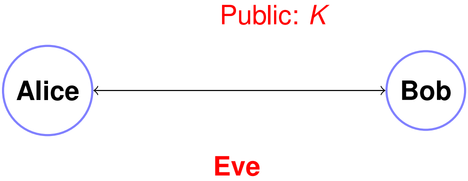
Disadvantages:

Shared secret!

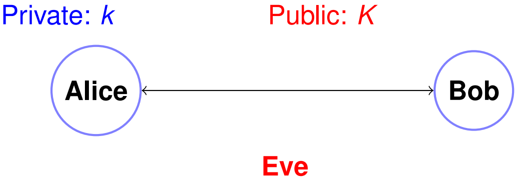
Uses up one time pad..or less and less secure.

Public key crypography.

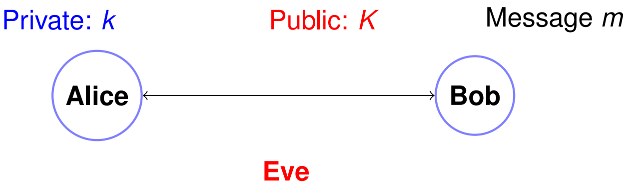
Public key crypography.



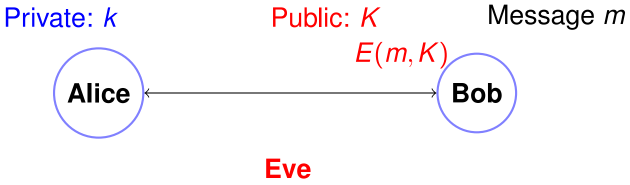
Public key crypography.



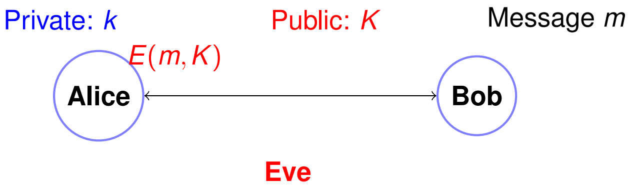
Public key crypography.



Public key crypography.

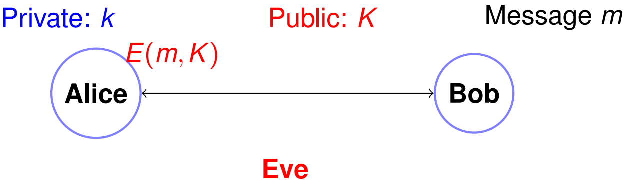


Public key crypography.



Public key crypography.

$m = D(E(m,K),k)$



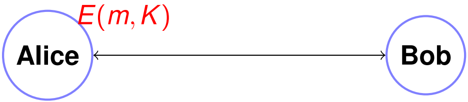
Public key crypography.

$m = D(E(m,K),k)$

Private: k

Public: K

Message m

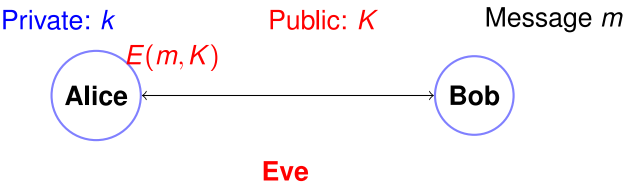


Eve

Everyone knows key K!

Public key crypography.

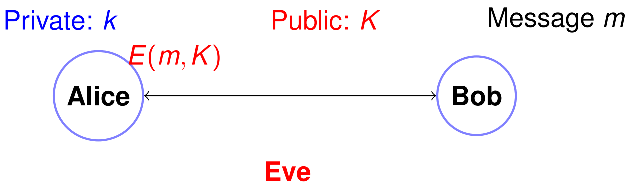
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve

Public key crypography.

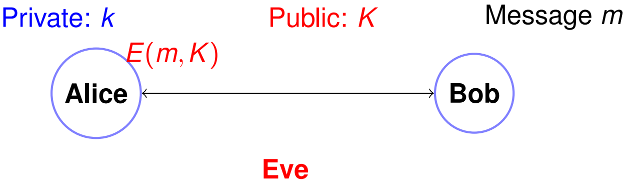
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve and me

Public key crypography.

$m = D(E(m,K),k)$



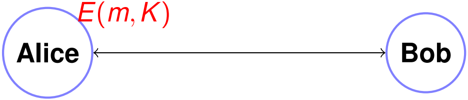
Everyone knows key K!
Bob (and Eve and me and you

Public key crypography.

$m = D(E(m,K),k)$

Private: k Public: K Message m

$E(m,K)$

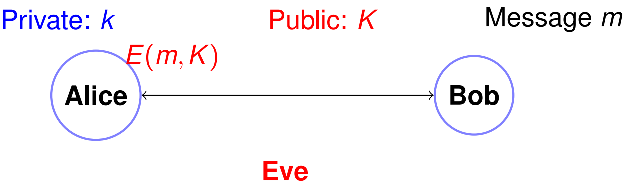


Eve

Everyone knows key K!
Bob (and Eve and me and you and you ...) can encode.

Public key crypography.

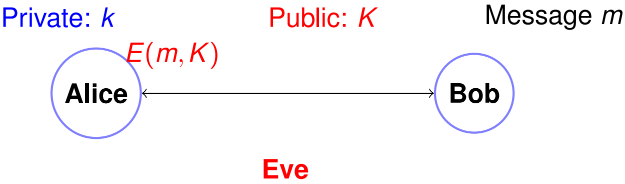
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve and me and you and you ...) can encode.
Only Alice knows the secret key k for public key K.

Public key crypography.

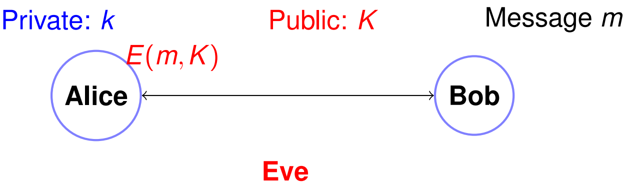
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve and me and you and you ...) can encode.
Only Alice knows the secret key k for public key K.
(Only?) Alice can decode with k.

Public key crypography.

$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve and me and you and you ...) can encode.
Only Alice knows the secret key k for public key K.
(Only?) Alice can decode with k.

Is this even possible?

Is public key crypto possible?

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system.

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow?

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.

---

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to "break" system. Too slow. Does it have to be slow? We don't really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N(= p \cdot q)$ and e: $K = (N, e)$ is my public key!

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to "break" system. Too slow. Does it have to be slow? We don't really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N(= p \cdot q)$ and e: $K = (N, e)$ is my public key!

Encoding: $\mod (x^e, N)$.

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N (= p \cdot q)$ and e: $K = (N, e)$ is my public key!

Encoding: $\mod (x^e, N)$.

Decoding: $\mod (y^d, N)$.

---

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to “break” system. Too slow. Does it have to be slow? We don’t really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N(= p \cdot q)$ and e: $K = (N, e)$ is my public key!

Encoding: $\mod(x^e, N)$.

Decoding: $\mod(y^d, N)$.

Does $D(E(m)) = m^{ed} = m \mod N$?

[1] Typically small, say $e = 3$.

Is public key crypto possible?

No. In a sense. One can try every message to "break" system. Too slow. Does it have to be slow? We don't really know.
...but we do public-key cryptography constantly!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N(= p \cdot q)$ and e: $K = (N, e)$ is my public key!

Encoding: $\mod (x^e, N)$.

Decoding: $\mod (y^d, N)$.

Does $D(E(m)) = m^{ed} = m \mod N$?

Yes!

[1] Typically small, say $e = 3$.

Poll

What is a piece of RSA?

Bob has a key (N,e,d). Alice is good, Eve is evil.

Poll

What is a piece of RSA?

Bob has a key (N,e,d). Alice is good, Eve is evil.

(A) Eve knows e and N.
(B) Alice knows e and N.
(C) $ed = 1 \pmod{N-1}$
(D) Bob forgot p and q but can still decode?
(E) Bob knows d
(F) $ed = 1 \pmod{(p-1)(q-1)}$ if $N = pq$.

Poll

What is a piece of RSA?

Bob has a key (N,e,d). Alice is good, Eve is evil.

(A) Eve knows e and N.
(B) Alice knows e and N.
(C) $ed = 1 \pmod{N-1}$
(D) Bob forgot p and q but can still decode?
(E) Bob knows d
(F) $ed = 1 \pmod{(p-1)(q-1)}$ if $N = pq$.

(A), (B), (D), (E), (F)

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$7(0) + 60(1) = 60$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$\begin{aligned} 7(0)+60(1) &= 60 \\ 7(1)+60(0) &= 7 \end{aligned}$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$\begin{aligned} 7(0) + 60(1) &= 60 \\ 7(1) + 60(0) &= 7 \\ 7(-8) + 60(1) &= 4 \end{aligned}$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$\begin{aligned}
7(0)+60(1) &= 60 \\
7(1)+60(0) &= 7 \\
7(-8)+60(1) &= 4 \\
7(9)+60(-1) &= 3
\end{aligned}$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$
\begin{aligned}
7(0)+60(1) &= 60 \\
7(1)+60(0) &= 7 \\
7(-8)+60(1) &= 4 \\
7(9)+60(-1) &= 3 \\
7(-17)+60(2) &= 1
\end{aligned}
$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$
\begin{aligned}
7(0) + 60(1) &= 60 \\
7(1) + 60(0) &= 7 \\
7(-8) + 60(1) &= 4 \\
7(9) + 60(-1) &= 3 \\
7(-17) + 60(2) &= 1
\end{aligned}
$$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7, 60) = 1$.
egcd(7,60).

$$
\begin{aligned}
7(0)+60(1) &= 60 \\
7(1)+60(0) &= 7 \\
7(-8)+60(1) &= 4 \\
7(9)+60(-1) &= 3 \\
7(-17)+60(2) &= 1
\end{aligned}
$$

Confirm:

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$
\begin{aligned}
7(0) + 60(1) &= 60 \\
7(1) + 60(0) &= 7 \\
7(-8) + 60(1) &= 4 \\
7(9) + 60(-1) &= 3 \\
7(-17) + 60(2) &= 1
\end{aligned}
$$

Confirm: $-119 + 120 = 1$

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7,60) = 1$.
egcd(7,60).

$$\begin{aligned} 7(0)+60(1) &= 60 \\ 7(1)+60(0) &= 7 \\ 7(-8)+60(1) &= 4 \\ 7(9)+60(-1) &= 3 \\ 7(-17)+60(2) &= 1 \end{aligned}$$

Confirm: $-119+120 = 1$

$d = e^{-1} = -17 = 43 = \pmod{60}$

Encryption/Decryption Techniques.

Encryption/Decryption Techniques.

Public Key: $(77, 7)$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2)$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e = 2^7$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$

Encryption/Decryption Techniques.

Public Key: $(77,7)$
Message Choices: $\{0,\ldots,76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$
uh oh!

Encryption/Decryption Techniques.

Public Key: $(77,7)$
Message Choices: $\{0,\ldots,76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$
uh oh!

Obvious way: 43 multiplications.

Encryption/Decryption Techniques.

Public Key: $(77,7)$
Message Choices: $\{0,\ldots,76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$
uh oh!

Obvious way: 43 multiplications. Ouch.

Encryption/Decryption Techniques.

Public Key: $(77,7)$
Message Choices: $\{0,\ldots,76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$
uh oh!

Obvious way: 43 multiplications. Ouch.

In general, $O(N)$ or $O(2^n)$ multiplications!

Repeated squaring.

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
51^{43}

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1}$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} =$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} =$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2})$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2}) = 60 * 60 = 3600 \equiv 58 \pmod{77}$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2}) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^{8} =$

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51)*(51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2})*(51^{2}) = 60*60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4})*(51^{4})$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2}) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4}) * (51^{4}) = 58 * 58 = 3364 \equiv 53 \pmod{77}$

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^1 \equiv 51 \pmod{77}$
$51^2 = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^4 = (51^2) * (51^2) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^8 = (51^4) * (51^4) = 58 * 58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^8) * (51^8) = 53 * 53 = 2809 \equiv 37 \pmod{77}$

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51)*(51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2})*(51^{2}) = 60*60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4})*(51^{4}) = 58*58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^{8})*(51^{8}) = 53*53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16})*(51^{16}) = 37*37 = 1369 \equiv 60 \pmod{77}$

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^1 \equiv 51 \pmod{77}$
$51^2 = (51)*(51) = 2601 \equiv 60 \pmod{77}$
$51^4 = (51^2)*(51^2) = 60*60 = 3600 \equiv 58 \pmod{77}$
$51^8 = (51^4)*(51^4) = 58*58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^8)*(51^8) = 53*53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16})*(51^{16}) = 37*37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2}) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4}) * (51^{4}) = 58 * 58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^{8}) * (51^{8}) = 53 * 53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16}) * (51^{16}) = 37 * 37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

$51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} = (60) * (53) * (60) * (51) \equiv 2 \pmod{77}$.

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^8 \cdot 51^2 \cdot 51^1 \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^1$.?
$51^1 \equiv 51 \pmod{77}$
$51^2 = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^4 = (51^2) * (51^2) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^8 = (51^4) * (51^4) = 58 * 58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^8) * (51^8) = 53 * 53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16}) * (51^{16}) = 37 * 37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

$51^{32} \cdot 51^8 \cdot 51^2 \cdot 51^1 = (60) * (53) * (60) * (51) \equiv 2 \pmod{77}$.

Decoding got the message back!

Repeated squaring.

Notice: $43 = 32+8+2+1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51)*(51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2})*(51^{2}) = 60*60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4})*(51^{4}) = 58*58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^{8})*(51^{8}) = 53*53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16})*(51^{16}) = 37*37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

$51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} = (60)*(53)*(60)*(51) \equiv 2 \pmod{77}$.

Decoding got the message back!

Repeated Squaring took 9 multiplications

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$ or 101011 in binary.
$51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^{1}$.?
$51^{1} \equiv 51 \pmod{77}$
$51^{2} = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^{4} = (51^{2}) * (51^{2}) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^{8} = (51^{4}) * (51^{4}) = 58 * 58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^{8}) * (51^{8}) = 53 * 53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16}) * (51^{16}) = 37 * 37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

$51^{32} \cdot 51^{8} \cdot 51^{2} \cdot 51^{1} = (60) * (53) * (60) * (51) \equiv 2 \pmod{77}$.

Decoding got the message back!

Repeated Squaring took 9 multiplications versus 43.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Note: $y = 2\lfloor y/2 \rfloor + \mod(y,2)$.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Note: $y = 2\lfloor y/2 \rfloor + \mod(y,2)$.

$x^y = x^{2(\lfloor y/2 \rfloor) + \mod(y,2)} = (x^2)^{\lfloor y/2 \rfloor} x^{y \mod 2} \pmod{m}$.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Note: $y = 2\lfloor y/2 \rfloor + \mod(y, 2)$.

$x^y = x^{2(\lfloor y/2 \rfloor) + \mod(y,2)} = (x^2)^{\lfloor y/2 \rfloor} x^{y \mod 2} \pmod{m}$.

Induction:
Recursive call on x^2 and $\lfloor y/2 \rfloor$, returns $(x^2)^{\lfloor y/2 \rfloor}$.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Note: $y = 2\lfloor y/2 \rfloor + \mod(y,2)$.

$x^y = x^{2(\lfloor y/2 \rfloor) + \mod(y,2)} = (x^2)^{\lfloor y/2 \rfloor} x^{y \mod 2} \pmod{m}$.

Induction:
Recursive call on x^2 and $\lfloor y/2 \rfloor$, returns $(x^2)^{\lfloor y/2 \rfloor}$.

≤ 2 multiplications per recursive call.

Recursive version.

```
(define (power x y m)
  (if (= y 1)
    (mod x m)
    (let ((x-to-evened-y  (power (square x) (/ y 2) m)))
    (if (evenp y)
        x-to-evened-y
        (mod (* x x-to-evened-y) m )))))
```

Claim: Program correctly computes x^y.

Base: $x^1 = x \pmod{m}$.

Note: $y = 2\lfloor y/2 \rfloor + \mod(y,2)$.

$x^y = x^{2(\lfloor y/2 \rfloor) + \mod(y,2)} = (x^2)^{\lfloor y/2 \rfloor} x^{y \mod 2} \pmod{m}$.

Induction:
Recursive call on x^2 and $\lfloor y/2 \rfloor$, returns $(x^2)^{\lfloor y/2 \rfloor}$.

≤ 2 multiplications per recursive call.

Note: $\lfloor y/2 \rfloor$ is integer division.

Repeated Squaring: x^y

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute x^1,

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2,$

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4,$

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots,$

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1. Example:

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1. Example: $43 = 101011$ in binary.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:

$O(n)$ multiplications.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:

$O(n)$ multiplications.
$O(n^2)$ time per multiplication.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:

$O(n)$ multiplications.
$O(n^2)$ time per multiplication.
$\implies O(n^3)$ time.

Conclusion: $x^y \mod N$

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $$x^{43} = x^{32} * x^8 * x^2 * x^1.$$

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:

$O(n)$ multiplications.
$O(n^2)$ time per multiplication.
$\implies O(n^3)$ time.

Conclusion: $x^y \mod N$ takes $O(n^3)$ time.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m,(N,e)) = m^e \pmod{N}$.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

For 512 bits, a few hundred million operations.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

For 512 bits, a few hundred million operations.
Easy, peasey.

Decoding.

$$E(m,(N,e)) = m^e \pmod{N}.$$

Decoding.

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

Decoding.

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

Decoding.

$$E(m,(N,e)) = m^e \pmod{N}.$$
$$D(m,(N,d)) = m^d \pmod{N}.$$

$N = pq$

Decoding.

$$E(m,(N,e)) = m^e \pmod N.$$
$$D(m,(N,d)) = m^d \pmod N.$$

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Decoding.

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want:

Decoding.

$$E(m,(N,e)) = m^e \pmod{N}.$$
$$D(m,(N,d)) = m^d \pmod{N}.$$

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Always decode correctly?

$$E(m,(N,e)) = m^e \pmod{N}.$$

Always decode correctly?

$$E(m,(N,e)) = m^e \pmod{N}.$$
$$D(m,(N,d)) = m^d \pmod{N}.$$

Always decode correctly?

$$E(m,(N,e)) = m^e \pmod{N}.$$
$$D(m,(N,d)) = m^d \pmod{N}.$$

Always decode correctly?

$$E(m,(N,e)) = m^e \pmod{N}.$$
$$D(m,(N,d)) = m^d \pmod{N}.$$

$N = pq$

Always decode correctly?

$$E(m,(N,e)) = m^e \pmod N.$$
$$D(m,(N,d)) = m^d \pmod N.$$

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want:

Always decode correctly?

$E(m,(N,e)) = m^e \pmod N$.
$D(m,(N,d)) = m^d \pmod N$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod N$.

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod{p}$$

Always decode correctly?

$E(m,(N,e)) = m^e \pmod N$.
$D(m,(N,d)) = m^d \pmod N$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod N$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1.$

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod p$,

$$a^{p-1} \equiv 1 \pmod p.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod p \implies$$

Always decode correctly?

$E(m,(N,e)) = m^e \pmod N$.
$D(m,(N,d)) = m^d \pmod N$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod N$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod p$,

$$a^{p-1} \equiv 1 \pmod p.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod p \implies a^{k(p-1)+1}$$

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}.$
$D(m,(N,d)) = m^d \pmod{N}.$

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}.$

Want: $(m^e)^d = m^{ed} = m \pmod{N}.$

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1.$

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod{p} \implies a^{k(p-1)+1} = a \pmod{p}$$

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod{p} \implies a^{k(p-1)+1} = a \pmod{p}$$

versus $\quad a^{k(p-1)(q-1)+1} = a \pmod{pq}$.

Always decode correctly?

$E(m,(N,e)) = m^e \pmod{N}$.
$D(m,(N,d)) = m^d \pmod{N}$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod{N}$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

$$\implies a^{k(p-1)} \equiv 1 \pmod{p} \implies a^{k(p-1)+1} = a \pmod{p}$$

versus $\quad a^{k(p-1)(q-1)+1} = a \pmod{pq}$.

Similar, not same, but useful.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$a^{p-1} \equiv 1 \pmod{p}$.

Proof:

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$a^{p-1} \equiv 1 \pmod{p}$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Each of $2, \ldots (p-1)$ has an inverse modulo p,

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Each of $2, \ldots (p-1)$ has an inverse modulo p, solve to get...

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$a^{p-1} \equiv 1 \pmod{p}$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Each of $2, \ldots (p-1)$ has an inverse modulo p, solve to get...

$$a^{(p-1)} \equiv 1 \mod p.$$

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$a^{p-1} \equiv 1 \pmod{p}$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Each of $2, \ldots (p-1)$ has an inverse modulo p, solve to get...

$$a^{(p-1)} \equiv 1 \mod p.$$

$\square$

Poll

Mark what is true.

Poll

Mark what is true.

(A) $2^7 = 1 \mod 7$
(B) $2^6 = 1 \mod 7$
(C) $2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7$ are distinct mod 7.
(D) $2^1, 2^2, 2^3, 2^4, 2^5, 2^6$ are distinct mod 7
(E) $2^{15} = 2 \mod 7$
(F) $2^{15} = 1 \mod 7$

Poll

Mark what is true.

(A) $2^7 = 1 \mod 7$
(B) $2^6 = 1 \mod 7$
(C) $2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7$ are distinct mod 7.
(D) $2^1, 2^2, 2^3, 2^4, 2^5, 2^6$ are distinct mod 7
(E) $2^{15} = 2 \mod 7$
(F) $2^{15} = 1 \mod 7$

(B), (F)

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Lemma 1: For any prime p and any a, b,
$a^{1+b(p-1)} \equiv a \pmod{p}$
Proof:

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Lemma 1: For any prime p and any a, b,
$a^{1+b(p-1)} \equiv a \pmod{p}$
Proof: If $a \equiv 0 \pmod{p}$, of course.

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Lemma 1: For any prime p and any a, b,
$a^{1+b(p-1)} \equiv a \pmod{p}$
Proof: If $a \equiv 0 \pmod{p}$, of course.

Otherwise
$a^{1+b(p-1)} \equiv$

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Lemma 1: For any prime p and any a, b,
$a^{1+b(p-1)} \equiv a \pmod{p}$
Proof: If $a \equiv 0 \pmod{p}$, of course.

Otherwise
$a^{1+b(p-1)} \equiv a^1 * (a^{p-1})^b$

Always decode correctly? (cont.)

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$$a^{p-1} \equiv 1 \pmod{p}.$$

Lemma 1: For any prime p and any a, b,
$a^{1+b(p-1)} \equiv a \pmod{p}$
Proof: If $a \equiv 0 \pmod{p}$, of course.

Otherwise
$a^{1+b(p-1)} \equiv a^1 * (a^{p-1})^b \equiv a * (1)^b \equiv a \pmod{p}$

$\square$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,
$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$$ $x^{1+k(q-1)(p-1)} - x$ is multiple of p and q.

...Decoding correctness...

Lemma 1: For any prime p and any a, b,

$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.

$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$ $x^{1+k(q-1)(p-1)} - x$ is multiple of p and q.

$$x^{1+k(q-1)(p-1)} - x \equiv 0 \mod (pq)$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,
$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.

$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$ $x^{1+k(q-1)(p-1)} - x$ is multiple of p and q.

$$x^{1+k(q-1)(p-1)} - x \equiv 0 \mod (pq) \implies x^{1+k(q-1)(p-1)} = x \mod pq.$$

...Decoding correctness...

Lemma 1: For any prime p and any a, b,
$$a^{1+b(p-1)} \equiv a \pmod{p}$$

Lemma 2: For any two different primes p, q and any x, k,
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

Let $a = x$, $b = k(p-1)$ and apply Lemma 1 with modulus q.
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{q}$$

Let $a = x$, $b = k(q-1)$ and apply Lemma 1 with modulus p.
$$x^{1+k(p-1)(q-1)} \equiv x \pmod{p}$$ $x^{1+k(q-1)(p-1)} - x$ is multiple of p and q.
$$x^{1+k(q-1)(p-1)} - x \equiv 0 \mod (pq) \implies x^{1+k(q-1)(p-1)} = x \mod pq.$$
$\square$

From CRT: $y = x \pmod{p}$ and $y = x \pmod{q} \implies y = x$.

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,

$$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \pmod{pq},$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,

$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \qquad \pmod{pq},$$

where $ed \equiv 1 \mod (p-1)(q-1) \implies ed = 1 + k(p-1)(q-1)$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \qquad \pmod{pq},$$

where $ed \equiv 1 \mod (p-1)(q-1) \implies ed = 1 + k(p-1)(q-1)$

$$x^{ed} \equiv$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \qquad \pmod{pq},$$

where $ed \equiv 1 \mod (p-1)(q-1) \implies ed = 1 + k(p-1)(q-1)$

$$x^{ed} \equiv x^{k(p-1)(q-1)+1}$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \qquad \pmod{pq},$$

where $ed \equiv 1 \mod (p-1)(q-1) \implies ed = 1 + k(p-1)(q-1)$

$$x^{ed} \equiv x^{k(p-1)(q-1)+1} \equiv x \pmod{pq}.$$

RSA decodes correctly..

Lemma 2: For any two different primes p, q and any x, k,
$x^{1+k(p-1)(q-1)} \equiv x \pmod{pq}$

Theorem: RSA correctly decodes!
Recall

$$D(E(x)) = (x^e)^d = x^{ed} \equiv x \pmod{pq},$$

where $ed \equiv 1 \mod (p-1)(q-1) \implies ed = 1 + k(p-1)(q-1)$

$$x^{ed} \equiv x^{k(p-1)(q-1)+1} \equiv x \pmod{pq}.$$

$\square$

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime?

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test..

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

2. Choose e with $\gcd(e,(p-1)(q-1)) = 1$.

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

2. Choose e with $\gcd(e,(p-1)(q-1)) = 1$.
   Use gcd algorithm to test.

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

2. Choose e with $\gcd(e,(p-1)(q-1)) = 1$.
   Use gcd algorithm to test.

3. Find inverse d of e modulo $(p-1)(q-1)$.

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

2. Choose e with $\gcd(e,(p-1)(q-1)) = 1$.
   Use gcd algorithm to test.

3. Find inverse d of e modulo $(p-1)(q-1)$.
   Use extended gcd algorithm.

Construction of keys.. ..

1. Find large (100 digit) primes p and q?

   Prime Number Theorem: $\pi(N)$ number of primes less than N.For all $N \geq 17$

   $$\pi(N) \geq N/\ln N.$$

   Choosing randomly gives approximately $1/(\ln N)$ chance of number being a prime. (How do you tell if it is prime? ... cs170..Miller-Rabin test.. Primes in P).

   For 1024 bit number, 1 in 710 is prime.

2. Choose e with $\gcd(e,(p-1)(q-1)) = 1$.
   Use gcd algorithm to test.

3. Find inverse d of e modulo $(p-1)(q-1)$.
   Use extended gcd algorithm.

All steps are polynomial in $O(\log N)$, the number of bits.

Security of RSA.

Security of RSA.

Security?

1. Alice knows p and q.
2. Bob only knows, $N(= pq)$, and e.

Security of RSA.

Security?

1. Alice knows p and q.
2. Bob only knows, $N(= pq)$, and e.

   Does not know, for example, d or factorization of N.

Security of RSA.

Security?

1. Alice knows p and q.
2. Bob only knows, $N(= pq)$, and e.
   Does not know, for example, d or factorization of N.
3. I don't know how to break this scheme without factoring N.

Security of RSA.

Security?

1. Alice knows p and q.
2. Bob only knows, $N(= pq)$, and e.
   Does not know, for example, d or factorization of N.
3. I don't know how to break this scheme without factoring N.

No one I know or have heard of admits to knowing how to factor N.

Security of RSA.

Security?

1. Alice knows p and q.
2. Bob only knows, $N(=pq)$, and e.

   Does not know, for example, d or factorization of N.
3. I don't know how to break this scheme without factoring N.

No one I know or have heard of admits to knowing how to factor N.
Breaking in general sense $\implies$ factoring algorithm.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

One trick:
Bob encodes credit card number, c,

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

One trick:
Bob encodes credit card number, c,
concatenated with random k-bit number r.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

One trick:
Bob encodes credit card number, c,
concatenated with random k-bit number r.

Never sends just c.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

One trick:
Bob encodes credit card number, c,
concatenated with random k-bit number r.

Never sends just c.

Again, more work to do to get entire system.

Much more to it.....

If Bobs sends a message (Credit Card Number) to Alice,

Eve sees it.

Eve can send credit card again!!

The protocols are built on RSA but more complicated;

For example, several rounds of challenge/response.

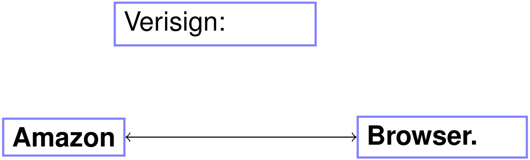
One trick:
  Bob encodes credit card number, c,
    concatenated with random k-bit number r.

Never sends just c.

Again, more work to do to get entire system.

CS161...

Signatures using RSA.



Signatures using RSA.

Verisign:

Amazon $\longleftrightarrow$ **Browser.**

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Signatures using RSA.

Verisign: k_v, K_v

Amazon $\longleftrightarrow$ **Browser.**

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Signatures using RSA.

$\boxed{\text{Verisign: } k_v, K_v}$

$\boxed{\textbf{Amazon}} \longleftrightarrow \boxed{\textbf{Browser. } K_v}$

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ ($N = pq$.)

Browser "knows" Verisign's public key: K_V.

Signatures using RSA.

Verisign: k_v, K_v

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$

$C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V)$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ ($N = pq$.)

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de}$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\quad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de} = C \pmod N$$

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ ($N = pq$.)

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de} = C \pmod N$

Valid signature of Amazon certificate C!

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$ $\qquad$ $C = E(S_V(C), k_V)$?

$[C, S_v(C)]$

Amazon $\longleftrightarrow$ **Browser.** K_v

Certificate Authority: Verisign, GoDaddy, DigiNotar,...

Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

Browser "knows" Verisign's public key: K_V.

Amazon Certificate: C = "I am Amazon. My public Key is K_A."

Versign signature of C: $S_v(C)$: $D(C, k_V) = C^d \mod N$.

Browser receives: $[C, y]$

Checks $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de} = C \pmod N$

Valid signature of Amazon certificate C!

Security: Eve can't forge unless she "breaks" RSA scheme.

RSA

RSA

Public Key Cryptography:

RSA

Public Key Cryptography:

$D(E(m,K),k) = (m^e)^d \mod N = m$.

RSA

Public Key Cryptography:

$D(E(m, K), k) = (m^e)^d \mod N = m$.

Signature scheme:

RSA

Public Key Cryptography:

$D(E(m,K),k) = (m^e)^d \mod N = m$.

Signature scheme:

$E(D(C,k),K) = (C^d)^e \mod N = C$

Poll

Signature authority has public key (N,e).

Poll

Signature authority has public key (N,e).

(A) Given message/signature (x,y) : check $y^d = x \pmod N$
(B) Given message/signature (x, y): check $y^e = x \pmod N$
(C) Signature of message x is $x^e \pmod N$
(D) Signature of message x is $x^d \pmod N$

Poll

Signature authority has public key (N,e).

(A) Given message/signature (x,y) : check $y^d = x \pmod N$
(B) Given message/signature (x, y): check $y^e = x \pmod N$
(C) Signature of message x is $x^e \pmod N$
(D) Signature of message x is $x^d \pmod N$

Other Eve.

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

2001..Doh.

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

2001..Doh.

... and August 28, 2011 announcement.

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

2001..Doh.

... and August 28, 2011 announcement.

DigiNotar Certificate issued for Microsoft!!!

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

2001..Doh.

... and August 28, 2011 announcement.

DigiNotar Certificate issued for Microsoft!!!

How does Microsoft get a CA to issue certificate to them ...

Other Eve.

Get CA to certify fake certificates: Microsoft Corporation.

2001..Doh.

... and August 28, 2011 announcement.

DigiNotar Certificate issued for Microsoft!!!

How does Microsoft get a CA to issue certificate to them ...

and only them?

Summary.

Public-Key Encryption.

Summary.

Public-Key Encryption.

RSA Scheme:

Summary.

Public-Key Encryption.

RSA Scheme:
$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.
  $E(x) = x^e \pmod{N}$.
  $D(y) = y^d \pmod{N}$.

Summary.

Public-Key Encryption.

RSA Scheme:
$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.
$E(x) = x^e \pmod{N}$.
$D(y) = y^d \pmod{N}$.

Repeated Squaring $\implies$ efficiency.

Summary.

Public-Key Encryption.

RSA Scheme:
$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.
$E(x) = x^e \pmod{N}$.
$D(y) = y^d \pmod{N}$.

Repeated Squaring $\implies$ efficiency.

Fermat's Theorem $\implies$ correctness.

Summary.

Public-Key Encryption.

RSA Scheme:
$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.
$E(x) = x^e \pmod{N}$.
$D(y) = y^d \pmod{N}$.

Repeated Squaring $\implies$ efficiency.

Fermat's Theorem $\implies$ correctness.

Good for Encryption

Summary.

Public-Key Encryption.

RSA Scheme:
$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.
  $E(x) = x^e \pmod{N}$.
  $D(y) = y^d \pmod{N}$.

Repeated Squaring $\implies$ efficiency.

Fermat's Theorem $\implies$ correctness.

Good for Encryption and Signature Schemes.