CS70: Lecture 11. Outline.

1. RSA system (continued)
   1.1 Correctness: Fermat's Theorem.
   1.2 Construction.
2. Signature Schemes.
3. Warnings.

Bijections

Bijection is **one to one** and **onto.**
Bijection:

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.

Bijections

Bijection is **one to one** and **onto.**

Bijection:

$f : A \to B$.

Domain: A, Co-Domain: B.

Versus Range.

E.g. **sin** (x).

Bijections

Bijection is **one to one** and **onto.**
Bijection:
  $f : A \to B$.
Domain: A, Co-Domain: B.
 Versus Range.
E.g. **sin** (x).
  $A = B =$ reals.

Bijections

Bijection is **one to one** and **onto.**

Bijection:

$f : A \to B$.

Domain: A, Co-Domain: B.

Versus Range.

E.g. **sin** (x).

$A = B =$ reals.

Range is $[-1, 1]$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1, 1]$. Onto: $[-1, 1]$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1, 1]$. Onto: $[-1, 1]$.
Not one-to-one.

Bijections

Bijection is **one to one** and **onto.**

Bijection:

$f : A \rightarrow B$.

Domain: A, Co-Domain: B.

Versus Range.

E.g. $\mathbf{sin}\,(x)$.

$A = B =$ reals.

Range is $[-1, 1]$. Onto: $[-1, 1]$.

Not one-to-one. $\mathbf{sin}\,(\pi) = \mathbf{sin}\,(0)$ = 0.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. $\mathbf{sin}\,(x)$.
$A = B =$ reals.
Range is $[-1, 1]$. Onto: $[-1, 1]$.
Not one-to-one. $\mathbf{sin}\,(\pi) = \mathbf{sin}\,(0) = 0$.

Range Definition always is onto.

Bijections

Bijection is **one to one** and **onto.**

Bijection:

$f : A \to B$.

Domain: A, Co-Domain: B.

Versus Range.

E.g. **sin** (x).

$A = B =$ reals.

Range is $[-1, 1]$. Onto: $[-1, 1]$.

Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.

Consider $f(x) = ax \mod m$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. $\mathbf{sin}\,(x)$.
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. $\mathbf{sin}\,(\pi) = \mathbf{sin}\,(0)$ = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1, 1]$. Onto: $[-1, 1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0, \ldots, m-1\} \to \{0, \ldots, m-1\}$.
Domain/Co-Domain: $\{0, \ldots, m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** $(0) = 0$.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

Bijections

Bijection is **one to one** and **onto.**

Bijection:

$f : A \to B$.

Domain: A, Co-Domain: B.

Versus Range.

E.g. **sin** (x).

$A = B =$ reals.

Range is $[-1,1]$. Onto: $[-1,1]$.

Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.

Consider $f(x) = ax \mod m$.

$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.

Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?
When $gcd(a,m)$ is ....

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?
When $gcd(a,m)$ is ....?

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?
When $gcd(a,m)$ is ....? ... 1.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?
When $gcd(a,m)$ is ....? ... 1.

Not Example: $a = 2$, $m = 4$,

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** (0) = 0.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0,\ldots,m-1\} \to \{0,\ldots,m-1\}$.
Domain/Co-Domain: $\{0,\ldots,m-1\}$.

Note: Why? Inverse if and only if $f(\cdot)$ one to one. Same size.

When is it a bijection?
When $gcd(a,m)$ is ....? ... 1.

Not Example: $a = 2$, $m = 4$, $f(0) = f(2) = 0 \pmod 4$.

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

Isomorphisms.

Bijection:

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

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

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

Isomorphisms.

Bijection:

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

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

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

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

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

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

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

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

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

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

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

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: $\quad (a,b) + (a',b') = (0,2)$.

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

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

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

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

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

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

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

Isomorphism:

Isomorphisms.

Bijection:

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

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

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

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

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

Isomorphisms.

Bijection:

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

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

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

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

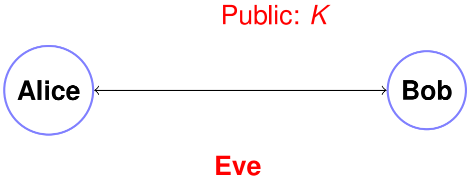
What is x where $x = 0 \pmod{5}$ and $x = 2 \pmod{9}$?

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

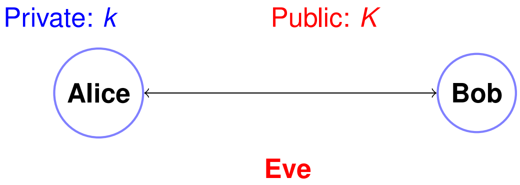
Isomorphism:
the actions under $\pmod{5}$, $\pmod{9}$
correspond to actions in $\pmod{45}$!

Public key crypography.

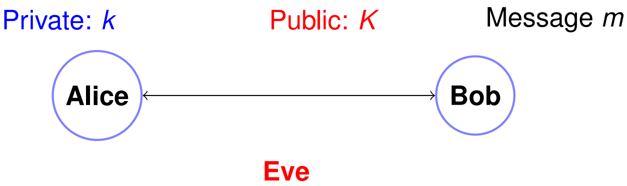
Public key crypography.



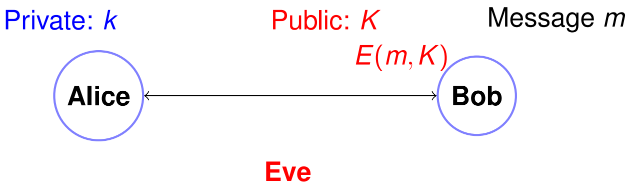
Public key crypography.



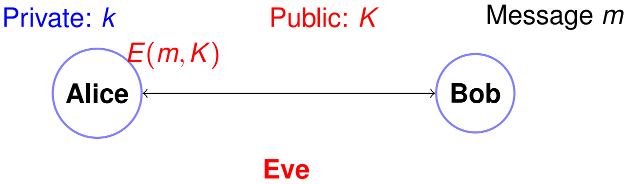
Public key crypography.



Public key crypography.

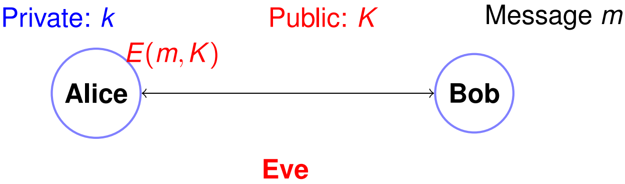


Public key crypography.



Public key crypography.

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

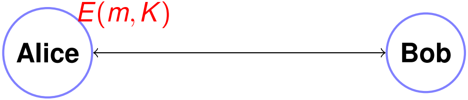


Public key crypography.

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

Private: k Public: K Message m

$E(m,K)$

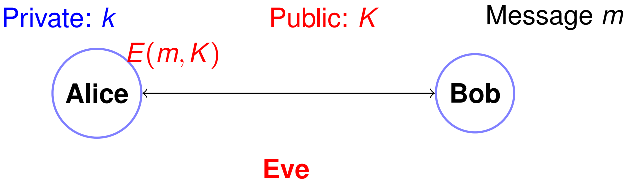


Eve

Everyone knows key K!

Public key crypography.

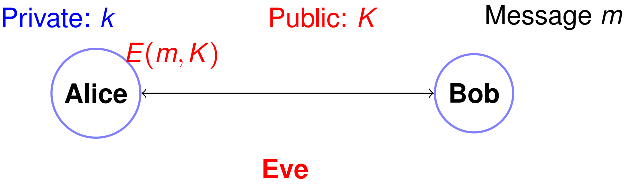
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve

Public key crypography.

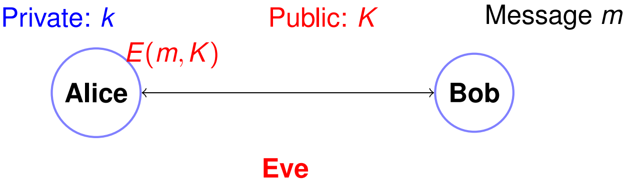
$m = D(E(m,K),k)$



Everyone knows key K!
Bob (and Eve and me

Public key crypography.

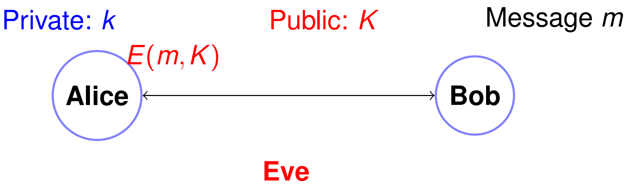
$m = D(E(m,K),k)$



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

Public key crypography.

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

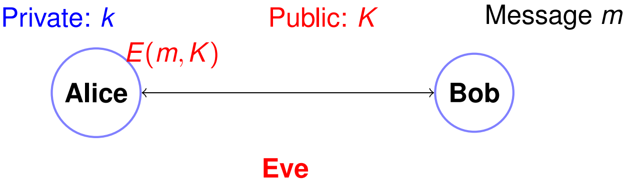


Everyone knows key K!

Bob (and Eve and me and you and you ...) can encode.

Public key crypography.

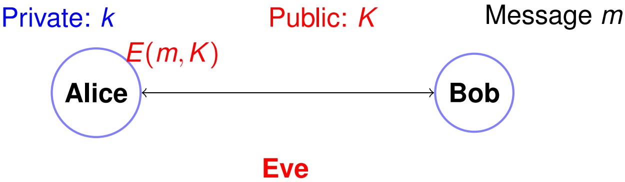
$m = D(E(m,K),k)$



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.

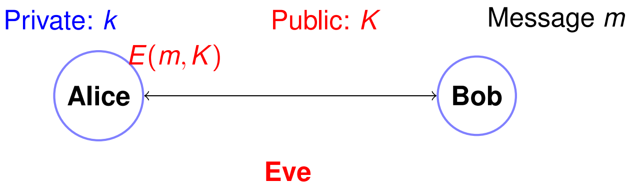
$m = D(E(m,K),k)$



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?

We don't really know.

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

Is public key crypto possible?

We don’t really know.
...but we do it every day!!!

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

Is public key crypto possible?

We don't really know.
...but we do it every day!!!

RSA (Rivest, Shamir, and Adleman)

---

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

Is public key crypto possible?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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?

We don't really know.
...but we do it every day!!!

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 $e = 3$.

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

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.

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

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...

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

$\square$

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.

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

...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.

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

$\square$

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

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

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.

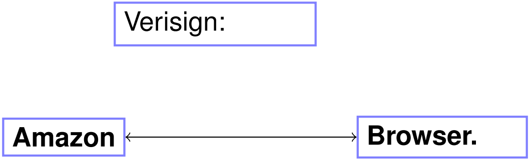
One trick:
  Bob encodes credit card number, c,
    concatenated with random k-bit number r.

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.

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.

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,...

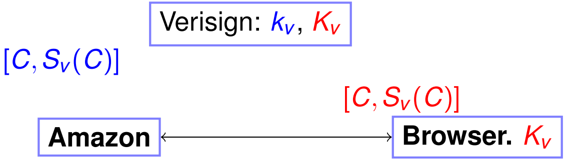
Verisign's key: $K_V = (N, e)$ and $k_V = d$ $(N = pq.)$

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.



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

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

$[C, S_v(C)]$

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 $E(y, K_V) = C$?

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

$[C, S_v(C)]$

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

$[C, S_v(C)]$

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

$[C, S_v(C)]$

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

$[C, S_v(C)]$

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

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

$[C, S_v(C)]$

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 $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de}$

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 $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de} = C \pmod N$

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 $E(y, K_V) = C$?

$E(S_v(C), K_V) = (S_v(C))^e = (C^d)^e = C^{de} = C \pmod N$

Valid signature of Amazon certificate C!

Signatures using RSA.

Verisign: k_v, K_v

$[C, S_v(C)]$

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

$[C, S_v(C)]$

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:

$E(D(C,k),K) = (C^d)^e \mod N = C$

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.