BILL, RECORD LECTURE!!!!

BILL RECORD LECTURE!!!

Public Key Cryptography: RSA

From The Economist Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

From **The Economist** Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

Context: In 2016 FARC, a left-wing insurgent group in Columbia, signed a peace treaty that ended 50 years of conflict **Yeah**!

From The Economist Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

Context: In 2016 FARC, a left-wing insurgent group in Columbia, signed a peace treaty that ended 50 years of conflict **Yeah**!

The former president of Columbia got the Nobel Peace Prize (the leader of FARC did not – I do not know why).

From The Economist Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

Context: In 2016 FARC, a left-wing insurgent group in Columbia, signed a peace treaty that ended 50 years of conflict **Yeah**!

The former president of Columbia got the Nobel Peace Prize (the leader of FARC did not – I do not know why).

However a more extreme insurgent group, ELN, is still active. Why did FARC negotiate but ELN did not?:

From **The Economist** Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

Context: In 2016 FARC, a left-wing insurgent group in Columbia, signed a peace treaty that ended 50 years of conflict **Yeah**!

The former president of Columbia got the Nobel Peace Prize (the leader of FARC did not – I do not know why).

However a more extreme insurgent group, ELN, is still active. Why did FARC negotiate but ELN did not?:

Quote: . . . And the ELN's **strong encryption system** has prevented the army from extracting information from seized computers, as it did with FARC.

From The Economist Sept 15, 2018, page 34

Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.

Context: In 2016 FARC, a left-wing insurgent group in Columbia, signed a peace treaty that ended 50 years of conflict **Yeah**!

The former president of Columbia got the Nobel Peace Prize (the leader of FARC did not – I do not know why).

However a more extreme insurgent group, ELN, is still active. Why did FARC negotiate but ELN did not?:

Quote: . . . And the ELN's **strong encryption system** has prevented the army from extracting information from seized computers, as it did with FARC.

Caveat: The article did not say what system they used. **Oh Well.**

Public Key Cryptography: RSA

What does RSA Stand For?

RSA stands for

What does RSA Stand For?

RSA stands for

Rivest-Shamir-Adelman.

What does RSA Stand For?

RSA stands for

Rivest-Shamir-Adelman.

They are the ones who came up with this cryptosystem.

Recall that DH was not · · ·

Diffie Hellman allowed Alice and Bob to share a secret string.

Recall that DH was not · · ·

Diffie Hellman allowed Alice and Bob to share a secret string.

Diffie Hellman *is not* an encryption system.

Recall that DH was not · · ·

Diffie Hellman allowed Alice and Bob to share a secret string.

Diffie Hellman *is not* an encryption system.

El Gamal *is* an encryption system but hard to use since its a 1-shot. You need to keep on doing DH to use it.

Recall that DH was not · · ·

Diffie Hellman allowed Alice and Bob to share a secret string.

Diffie Hellman *is not* an encryption system.

El Gamal *is* an encryption system but hard to use since its a 1-shot. You need to keep on doing DH to use it.

RSA is an encryption system.

Slight Variant on Fermat's Little Theorem

Recall Fermat's little Theorem
Thm If p is prime and $a \in \mathbb{N}$ then

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

Slight Variant on Fermat's Little Theorem

Recall Fermat's little Theorem
Thm If p is prime and $a \in \mathbb{N}$ then

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

We want to divide both sides by a and get $a^{p-1} \equiv 1 \pmod{p}$.

Slight Variant on Fermat's Little Theorem

Recall Fermat's little Theorem
Thm If p is prime and $a \in \mathbb{N}$ then

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

We want to divide both sides by a and get $a^{p-1} \equiv 1 \pmod{p}$.
Not quite right: What if $a \equiv 0 \pmod{p}$? Then not true. Hence:

Slight Variant on Fermat's Little Theorem

Recall Fermat's little Theorem

Thm If p is prime and $a \in \mathbb{N}$ then

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

We want to divide both sides by a and get $a^{p-1} \equiv 1 \pmod{p}$.

Not quite right: What if $a \equiv 0 \pmod{p}$? Then not true. Hence:

Thm If p is prime and $a \in \mathbb{N}$ and $a \not\equiv 0 \pmod{p}$ then

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

Slight Variant on Fermat's Little Theorem

Recall Fermat's little Theorem

Thm If p is prime and $a \in \mathbb{N}$ then

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

We want to divide both sides by a and get $a^{p-1} \equiv 1 \pmod{p}$.

Not quite right: What if $a \equiv 0 \pmod{p}$? Then not true. Hence:

Thm If p is prime and $a \in \mathbb{N}$ and $a \not\equiv 0 \pmod{p}$ then

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

We will refer to both as **Fermat's Little Theorem.**

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.

$11^{999{,}999{,}999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$$11^{999,999,999} = 11^{106k} \times 11^{27} = (11^{106})^k \times 11^{27} \equiv 1^k 11^{27} \equiv 11^{27} \quad (\text{mod } 10$$

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$$11^{999,999,999} = 11^{106k} \times 11^{27} = (11^{106})^{k} \times 11^{27} \equiv 1^{k} 11^{27} \equiv 11^{27} \quad (\text{mod } 10$$

$$11^{999,999,999} \equiv 11^{999,999,999 \pmod{106}} \pmod{107} \equiv 11^{27} \pmod{107}$$

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$$11^{999,999,999} = 11^{106k} \times 11^{27} = (11^{106})^k \times 11^{27} \equiv 1^k 11^{27} \equiv 11^{27} \quad (\text{mod } 10$$

$$11^{999,999,999} \equiv 11^{999,999,999 \pmod{106}} \pmod{107} \equiv 11^{27} \pmod{107}$$

Now do normal repeated squaring, $2\lg(27) = 10$. Can do better.
Recall its really
$\lg(27)+$ the number of 1's in the binary rep of 27.

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$$11^{999,999,999} = 11^{106k} \times 11^{27} = (11^{106})^k \times 11^{27} \equiv 1^k 11^{27} \equiv 11^{27} \pmod{10$$

$$11^{999,999,999} \equiv 11^{999,999,999 \pmod{106}} \pmod{107} \equiv 11^{27} \pmod{107}$$

Now do normal repeated squaring, $2\lg(27) = 10$. Can do better.
Recall its really
$\lg(27)+$ the number of 1's in the binary rep of 27.
Can we generalize?

$11^{999,999,999} \pmod{107}$

Repeated squaring would take $\sim \lg(999,999,999) \sim 30$ mults.
By Fermat's Little Thm $11^{106} \equiv 1 \pmod{107}$.
Note
$999,999,999 \equiv 27 \pmod{106}$
Hence

$$999,999,999 = 106k + 27 \text{ (don't care what } k \text{ is)}$$

$$11^{999,999,999} = 11^{106k} \times 11^{27} = (11^{106})^k \times 11^{27} \equiv 1^k 11^{27} \equiv 11^{27} \pmod{10$$

$$11^{999,999,999} \equiv 11^{999,999,999 \pmod{106}} \pmod{107} \equiv 11^{27} \pmod{107}$$

Now do normal repeated squaring, $2\lg(27) = 10$. Can do better.
Recall its really
$\lg(27) +$ the number of 1's in the binary rep of 27.
Can we generalize? **Yes**

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

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

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

$m = k(p-1) + r$

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

$m = k(p-1) + r$

Hence:

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

$m = k(p-1) + r$

Hence:

$$a^m \equiv a^{k(p-1)+r} \equiv (a^{p-1})^k \times a^r \equiv 1^k a^r \equiv a^r$$

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

$m = k(p-1) + r$

Hence:

$$a^m \equiv a^{k(p-1)+r} \equiv (a^{p-1})^k \times a^r \equiv 1^k a^r \equiv a^r$$

Since $r \equiv m \pmod{p-1}$, $a^m \equiv a^{m \bmod p-1} \pmod{p}$

Exponentiation with Really Big Exponents

Generalize p prime, $a \not\equiv 0 \pmod{p}$, $m \in \mathbb{N}$.

We want to compute $a^m \pmod{p}$.

We know that $a^{p-1} \equiv 1 \pmod{p}$. Divide m by $p-1$:

$m = k(p-1) + r$

Hence:

$$a^m \equiv a^{k(p-1)+r} \equiv (a^{p-1})^k \times a^r \equiv 1^k a^r \equiv a^r$$

Since $r \equiv m \pmod{p-1}$, $a^m \equiv a^{m \bmod p-1} \pmod{p}$

This last equation is the important point

Needed Mathematics- The ϕ Function

Next few slides are on the ϕ function.

Needed Mathematics- The ϕ Function

Next few slides are on the ϕ function.

YES, you have already seen it.

Needed Mathematics- The ϕ Function

Next few slides are on the ϕ function.

YES, you have already seen it.

As the saying goes:
Math is best learned twice... at least twice.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \leq a \leq p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \le a \le p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.

Recall For all m, $a^m \equiv a^{m \pmod{p-1}} \pmod{p}$.

So arithmetic in the exponents is mod $p-1$.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \le a \le p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.

Recall For all m, $a^m \equiv a^{m \pmod{p-1}} \pmod{p}$.

So arithmetic in the exponents is mod $p-1$.

We need to generalize this to when the mod is **not** a prime.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \le a \le p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.
Recall For all m, $a^m \equiv a^{m \pmod{p-1}} \pmod{p}$.
So arithmetic in the exponents is mod $p-1$.

We need to generalize this to when the mod is **not** a prime.

Definition $\phi(n)$ is the number of numbers in $\{1, \ldots, n\}$ that are relatively prime to n.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \le a \le p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.
Recall For all m, $a^m \equiv a^{m \pmod{p-1}} \pmod{p}$.
So arithmetic in the exponents is mod $p-1$.

We need to generalize this to when the mod is **not** a prime.

Definition $\phi(n)$ is the number of numbers in $\{1, \ldots, n\}$ that are relatively prime to n.
Recall If p is prime then $\phi(p) = p-1$.

Needed Mathematics- The ϕ Function (cont)

Recall If p is prime and $1 \le a \le p-1$ then $a^{p-1} \equiv 1 \pmod{p}$.
Recall For all m, $a^m \equiv a^{m \pmod{p-1}} \pmod{p}$.
So arithmetic in the exponents is mod $p-1$.

We need to generalize this to when the mod is **not** a prime.

Definition $\phi(n)$ is the number of numbers in $\{1, \ldots, n\}$ that are relatively prime to n.
Recall If p is prime then $\phi(p) = p - 1$.
Recall If a, b rel prime then $\phi(ab) = \phi(a)\phi(b)$.

Theorem for Primes, Theorem for n

We restate and generalize.

Theorem for Primes, Theorem for n

We restate and generalize.
Fermat's Little Theorem If p is prime and $a \not\equiv 0 \pmod{p}$ then

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

Theorem for Primes, Theorem for n

We restate and generalize.

Fermat's Little Theorem If p is prime and $a \not\equiv 0 \pmod{p}$ then

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

Restate:

Fermat's Little Theorem If p is prime and a is rel prime to p then

$$a^m \equiv a^{m \bmod \phi(p)} \pmod{p}.$$

Theorem for Primes, Theorem for n

We restate and generalize.

Fermat's Little Theorem If p is prime and $a \not\equiv 0 \pmod{p}$ then

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

Restate:

Fermat's Little Theorem If p is prime and a is rel prime to p then

$$a^m \equiv a^{m \bmod \phi(p)} \pmod{p}.$$

Generalize:

Fermat-Euler Theorem If $n \in \mathbb{N}$ and a is rel prime to n then

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}.$$

Examples

$$14^{999,999} \pmod{393}$$

Examples

$$14^{999{,}999} \pmod{393}$$

$$\phi(393) = \phi(3 \times 131) = \phi(3) \times \phi(131) = 2 \times 130 = 260.$$

Examples

$$14^{999{,}999} \pmod{393}$$

$$\phi(393) = \phi(3 \times 131) = \phi(3) \times \phi(131) = 2 \times 130 = 260.$$

$$14^{999{,}999} = 14^{999{,}999 \pmod{260}} \pmod{393} \equiv 14^{39} \pmod{393}$$

Examples

$$14^{999,999} \pmod{393}$$

$$\phi(393) = \phi(3 \times 131) = \phi(3) \times \phi(131) = 2 \times 130 = 260.$$

$$14^{999,999} = 14^{999,999 \pmod{260}} \pmod{393} \equiv 14^{39} \pmod{393}$$

Now just do repeated squaring.

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191{,}992{,}194{,}299{,}292{,}777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191,992,194,299,292,777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.
This is true! There will be some HW using it.

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191,992,194,299,292,777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.
This is true! There will be some HW using it.

You are thinking

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191,992,194,299,292,777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.
This is true! There will be some HW using it.

You are thinking A&B will need to do $a^m \pmod{n}$ for **large** m.

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191,992,194,299,292,777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.
This is true! There will be some HW using it.

You are thinking A&B will need to do $a^m \pmod{n}$ for **large** m.

No. That is not what we will be doing, though I see why you would think that.

Bait and Switch

I got you interested in the theorem

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}$$

by telling you that it can be used to do things like

$$17^{191,992,194,299,292,777} \pmod{150}.$$

with **much less than** $2\lg(191,992,194,299,292,777)$ mults.
This is true! There will be some HW using it.

You are thinking A&B will need to do $a^m \pmod{n}$ for **large** m.

No. That is not what we will be doing, though I see why you would think that.
We will just use the theorem:

$$a^m \equiv a^{m \bmod \phi(n)} \pmod{n}.$$

Easy and Hard

Easy or Hard?

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q.

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$.

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.)

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$.

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$. **Easy.**

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$. **Easy.**
5. Given N, e find d such that $ed \equiv 1 \pmod{R}$.

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$. **Easy.**
5. Given N, e find d such that $ed \equiv 1 \pmod{R}$. **Hard.**

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$. **Easy.**
5. Given N, e find d such that $ed \equiv 1 \pmod{R}$. **Hard.**
6. Compute $m^e \pmod{N}$.

Easy and Hard

Easy or Hard?

1. Given L, generate two primes of length L: p, q. **Easy.**
2. Given p, q find $N = pq$ and $R = \phi(N) = (p-1)(q-1)$. **Easy.**
3. Given R find an e rel prime to R. (e for encrypt.) **Easy.**
4. Given R, e find d such that $ed \equiv 1 \pmod{R}$. **Easy.**
5. Given N, e find d such that $ed \equiv 1 \pmod{R}$. **Hard.**
6. Compute $m^e \pmod{N}$. **Easy.**

RSA

Let L be a security parameter

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.
7. If **Alice** gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \bmod R} \equiv m^{1 \bmod R} \equiv m \pmod{N}.$$

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.
7. If **Alice** gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \bmod R} \equiv m^{1 \bmod R} \equiv m \pmod{N}.$$

PRO Alice and Bob can execute the protocol easily.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.
7. If **Alice** gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \bmod R} \equiv m^{1 \bmod R} \equiv m \pmod{N}.$$

PRO Alice and Bob can execute the protocol easily.
Biggest PRO Alice and Bob never had to meet!

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.
7. If **Alice** gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \bmod R} \equiv m^{1 \bmod R} \equiv m \pmod{N}.$$

PRO Alice and Bob can execute the protocol easily.
Biggest PRO Alice and Bob never had to meet!
PRO Bob can control the message.

RSA

Let L be a security parameter

1. **Alice** picks two primes p, q of length L and computes $N = pq$.
2. **Alice** computes $R = \phi(N) = \phi(pq) = (p-1)(q-1)$.
3. **Alice** picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R.
4. **Alice** finds d such that $ed \equiv 1 \pmod{R}$.
5. **Alice** broadcasts (N, e). (Bob and Eve both see it.)
6. **Bob** To send $m \in \{1, \ldots, N-1\}$, broadcast $m^e \pmod{N}$.
7. If **Alice** gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \bmod R} \equiv m^{1 \bmod R} \equiv m \pmod{N}.$$

PRO Alice and Bob can execute the protocol easily.
Biggest PRO Alice and Bob never had to meet!
PRO Bob can control the message.
Question Can Eve find out m?

Convention for RSA

Alice sends (N, e) to get the process started.

Convention for RSA

Alice sends (N, e) to get the process started.

Then Bob can send Alice messages.

Convention for RSA

Alice sends (N, e) to get the process started.

Then Bob can send Alice messages.

We don't have Alice sending Bob messages.

Convention for RSA

Alice sends (N, e) to get the process started.

Then Bob can send Alice messages.

We don't have Alice sending Bob messages.

In examples we do in slides and HW we might not have $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ since we want to have easy computations for educational purposes.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.
$N = pq = 31 * 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.
$N = pq = 31 * 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$.
Use $e = 77$, e rel prime to R
Find $d = 533$ ($ed \equiv 1 \pmod{R}$)
Check $ed = 77 * 533 = 41041 \equiv 1 \pmod{1080}$.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.

$N = pq = 31 * 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$.
Use $e = 77$, e rel prime to R
Find $d = 533$ ($ed \equiv 1 \pmod{R}$)
Check $ed = 77 * 533 = 41041 \equiv 1 \pmod{1080}$.
Bob pick an $m \in \{1, \ldots, N-1\} = \{1, \ldots, 1146\}$. Do not tell us what it is.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.

$N = pq = 31 * 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$.
Use $e = 77$, e rel prime to R
Find $d = 533$ ($ed \equiv 1 \pmod{R}$)
Check $ed = 77 * 533 = 41041 \equiv 1 \pmod{1080}$.
Bob pick an $m \in \{1, \ldots, N-1\} = \{1, \ldots, 1146\}$. Do not tell us what it is.
Bob compute $c = m^e \pmod{1147}$ and tell it to us.

Do RSA in Class

Pick out two students to be Alice and Bob.
Use primes:
$p = 31$, Prime.
$q = 37$, Prime.
$N = pq = 31 * 37 = 1147.$
$R = \phi(N) = 30 * 36 = 1080.$
Use $e = 77$, e rel prime to R
Find $d = 533$ ($ed \equiv 1 \pmod{R}$)
Check $ed = 77 * 533 = 41041 \equiv 1 \pmod{1080}$.
Bob pick an $m \in \{1, \ldots, N-1\} = \{1, \ldots, 1146\}$. Do not tell us what it is.
Bob compute $c = m^e \pmod{1147}$ and tell it to us.
Alice compute $c^d \pmod{1147}$, should get back m.

How well the Class does on RSA-in-Class

I have taught 456 4 times (including Fall 2021) and so far

How well the Class does on RSA-in-Class

I have taught 456 4 times (including Fall 2021) and so far
3 out of the 4 times the students DID get the same answer!

How well the Class does on RSA-in-Class

I have taught 456 4 times (including Fall 2021) and so far
3 out of the 4 times the students DID get the same answer!

The one time they did not, Bob did m^e mod 1080 instead of 1147.

How well the Class does on RSA-in-Class

I have taught 456 4 times (including Fall 2021) and so far
3 out of the 4 times the students DID get the same answer!

The one time they did not, Bob did m^e mod 1080 instead of 1147.

In Fall 2021 Bob did this but caught her mistake before it lead to an error.

How well the Class does on RSA-in-Class

I have taught 456 4 times (including Fall 2021) and so far
3 out of the 4 times the students DID get the same answer!

The one time they did not, Bob did m^e mod 1080 instead of 1147.

In Fall 2021 Bob did this but caught her mistake before it lead to an error.

Wars have been lost due to errors like that that do not get detected.

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. $(p, q, R$ are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

If Factoring Easy then RSA is crackable

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

If Factoring Easy then RSA is crackable

What about converse?

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

If Factoring Easy then RSA is crackable

What about converse?

If RSA is crackable then Factoring is Easy

VOTE TRUE or FALSE or UNKNOWN TO SCIENCE

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

If Factoring Easy then RSA is crackable

What about converse?

If RSA is crackable then Factoring is Easy

VOTE TRUE or FALSE or UNKNOWN TO SCIENCE
UNKNOWN TO SCIENCE.

What Do We Really Know about RSA

If Eve can factor then she can crack RSA.

1. Input (N, e) where $N = pq$ and e is rel prime to $R = (p-1)(q-1)$. (p, q, R are NOT part of the input.)
2. Eve factors N to find p, q. Eve computes $R = (p-1)(q-1)$.
3. Eve finds d such that $ed \equiv 1 \pmod{R}$.

If Factoring Easy then RSA is crackable

What about converse?

If RSA is crackable then Factoring is Easy

VOTE TRUE or FALSE or UNKNOWN TO SCIENCE
UNKNOWN TO SCIENCE.
Note In ugrad math classes rare to have a statement that is **UNKNOWN TO SCIENCE**. **Discuss**.

Hardness Assumption

Definition Let $RSAF$ be the following function:
Input $N, e, m^e \pmod{N}$ (know $N = pq$ but don't know p, q).

Outputs m.

Hardness Assumption

Definition Let $RSAF$ be the following function:
Input $N, e, m^e \pmod{N}$ (know $N = pq$ but don't know p, q).

Outputs m.

Hardness assumption (HA) $RSAF$ is hard to compute.

Hardness Assumption

Definition Let $RSAF$ be the following function:
Input $N, e, m^e \pmod{N}$ (know $N = pq$ but don't know p, q).

Outputs m.

Hardness assumption (HA) $RSAF$ is hard to compute.

One can show, assuming HA that RSA is hard to crack. But this proof will depend on a model of security. See caveats about this on similar DH slides (bribery, timing attacks, Maginot Line).

What Could be True?

The following are all possible:

What Could be True?

The following are all possible:

1) Factoring easy. RSA is crackable.

What Could be True?

The following are all possible:

1) Factoring easy. RSA is crackable.

2) Factoring hard, HA false. RSA crackable, Factoring hard!!

What Could be True?

The following are all possible:

1) Factoring easy. RSA is crackable.

2) Factoring hard, HA false. RSA crackable, Factoring hard!!

3) Factoring hard, HA true, but RSA is crackable by other means, e.g., Timing Attacks. Must rethink our model of security.

What Could be True?

The following are all possible:

1) Factoring easy. RSA is crackable.

2) Factoring hard, HA false. RSA crackable, Factoring hard!!

3) Factoring hard, HA true, but RSA is crackable by other means, e.g., Timing Attacks. Must rethink our model of security.

4) Factoring hard, HA true, and RSA remains uncracked for years. Increases our confidence but . . ..

What Could be True?

The following are all possible:

1) Factoring easy. RSA is crackable.

2) Factoring hard, HA false. RSA crackable, Factoring hard!!

3) Factoring hard, HA true, but RSA is crackable by other means, e.g., Timing Attacks. Must rethink our model of security.

4) Factoring hard, HA true, and RSA remains uncracked for years. Increases our confidence but . . ..

Items 3 and 4 is current state with some caveats: Do Alice and Bob use it properly? Do they have large enough parameters? What is Eve's computing power?

Making RSA More Efficient

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!
Kunal finds an Automata-Theory-connection to that e!

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!
Kunal finds an Automata-Theory-connection to that e!
Josh finds an Algebraic-Geometry-connection to that e! etc.

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!
Kunal finds an Automata-Theory-connection to that e!
Josh finds an Algebraic-Geometry-connection to that e! etc.

Student: I've read on the web that you should use $e = 2^{2^4} + 1$, the fourth Fermat Prime. And the article **20 years of attacks on RSA** (on the course website now) says so. The article was written by a theorist like you, Dan Boneh.

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!
Kunal finds an Automata-Theory-connection to that e!
Josh finds an Algebraic-Geometry-connection to that e! etc.

Student: I've read on the web that you should use $e = 2^{2^4} + 1$, the fourth Fermat Prime. And the article **20 years of attacks on RSA** (on the course website now) says so. The article was written by a theorist like you, Dan Boneh.

Bill: Dan Boneh is a **much better theorist** than me. Email me the website and paper and I'll see whats up.

Debate Between Bill and Student (Fall 2019)

Bill: Alice should not use the same value of e all the time. If she does then that e becomes an object of study.
Sajjad finds a Ramsey-Theory-connection to that e!
Kunal finds an Automata-Theory-connection to that e!
Josh finds an Algebraic-Geometry-connection to that e! etc.

Student: I've read on the web that you should use $e = 2^{2^4} + 1$, the fourth Fermat Prime. And the article **20 years of attacks on RSA** (on the course website now) says so. The article was written by a theorist like you, Dan Boneh.

Bill: Dan Boneh is a **much better theorist** than me. Email me the website and paper and I'll see whats up.
Well pierce my ears and call me drafty! In practice you SHOULD use $e = 2^{2^4} + 1$.

Why is $e = 2^{2^4} + 1$ Good to Use?

Recall that in RSA Bob must compute m^e.

Why is $e = 2^{2^4} + 1$ Good to Use?

Recall that in RSA Bob must compute m^e.
Bill Can do m^e with repeated squaring in **roughly** $\lg_2(m)$ steps.

Why is $e = 2^{2^4} + 1$ Good to Use?

Recall that in RSA Bob must compute m^e.
Bill Can do m^e with repeated squaring in **roughly** $\lg_2(m)$ steps.

Practitioner **roughly** $\lg_2(m)$ steps? Don't give me BS words like **roughly**. Are you one of those **big-O** people where the constant is, like, a gazillion?

Why is $e = 2^{2^4} + 1$ Good to Use?

Recall that in RSA Bob must compute m^e.
Bill Can do m^e with repeated squaring in **roughly** $\lg_2(m)$ steps.

Practitioner **roughly** $\lg_2(m)$ steps? Don't give me BS words like **roughly**. Are you one of those **big-O** people where the constant is, like, a gazillion?

Bill I've been called worse. **Irene** recently emailed me a slide correction and called me a **donut-brained Squid**. I think that's an insult.

Why is $e = 2^{2^4} + 1$ Good to Use?

Recall that in RSA Bob must compute m^e.
Bill Can do m^e with repeated squaring in **roughly** $\lg_2(m)$ steps.

Practitioner **roughly** $\lg_2(m)$ steps? Don't give me BS words like **roughly**. Are you one of those **big-O** people where the constant is, like, a gazillion?

Bill I've been called worse. **Irene** recently emailed me a slide correction and called me a **donut-brained Squid**. I think that's an insult.

Practitioner Let compare using $e = 2^{2^4} + 1$ to using $e = 2^{2^4} - 1$.

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m.$$

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m.$

Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m$.
Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.
Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m.$

Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.

Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$2^{2^4} - 1 = m^{2^{2^4}-1} = m^{2^0} \times m^{2^1} \times \cdots m^{2^{2^4-1}}$

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m$.
Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.
Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$2^{2^4} - 1 = m^{2^{2^4}-1} = m^{2^0} \times m^{2^1} \times \cdots m^{2^{2^4-1}}$
Repeated Squaring: $m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4-1}}$. 15 steps.

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m.$

Repeated Squaring: $m^{2^0}, m^2,\ m^{2^2},\ m^{2^3},\ \ldots,\ m^{2^{2^4}}$. 16 steps.

Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$2^{2^4} - 1 = m^{2^{2^4}-1} = m^{2^0} \times m^{2^1} \times \cdots m^{2^{2^4-1}}$

Repeated Squaring: $m^2,\ m^{2^2},\ m^{2^3},\ \ldots,\ m^{2^{2^4-1}}$. 15 steps.

Then $m^2 \times m^{2^2} \times m^{2^3} \times \ldots \times m^{2^{2^4-1}}$. 15 **more** steps. **30 steps**.

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m.$

Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.

Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$2^{2^4} - 1 = m^{2^{2^4}-1} = m^{2^0} \times m^{2^1} \times \cdots m^{2^{2^4-1}}$

Repeated Squaring: $m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4-1}}$. 15 steps.

Then $m^2 \times m^{2^2} \times m^{2^3} \times \ldots \times m^{2^{2^4-1}}$. 15 **more** steps. **30 steps**.

Bill: Does **17** vs **30** steps matter?

$e = 2^{2^4} + 1$ vs $e = 2^{2^4} - 1$

$m^{2^{2^4}+1} = m^{2^{2^4}+1} = m^{2^{2^4}} \times m$.
Repeated Squaring: $m^{2^0}, m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4}}$. 16 steps.
Then $(m^{2^{2^4}}) \times m = m^{2^{2^4}+1}$. **17 steps total**.

$2^{2^4} - 1 = m^{2^{2^4}-1} = m^{2^0} \times m^{2^1} \times \cdots m^{2^{2^4-1}}$
Repeated Squaring: $m^2, m^{2^2}, m^{2^3}, \ldots, m^{2^{2^4-1}}$. 15 steps.
Then $m^2 \times m^{2^2} \times m^{2^3} \times \ldots \times m^{2^{2^4-1}}$. 15 **more** steps. **30 steps**.

Bill: Does **17** vs **30** steps matter?

Practitioner: Yes, duh. It's almost twice as fast!

$e = 2^{2^4} + 1$ vs My Fears

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

In Theory: Do not want to use **the same** e over and over again for fear of this being exploited.

Who is Right: $e = 2^{16} + 1$ is used a lot.

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

In Theory: Do not want to use **the same** e over and over again for fear of this being exploited.

Who is Right: $e = 2^{16} + 1$ is used a lot. Should it be?

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

In Theory: Do not want to use **the same** e over and over again for fear of this being exploited.

Who is Right: $e = 2^{16} + 1$ is used a lot. Should it be?

- Nobody in academia has cracked RSA just using that $e = 2^{2^4} - 1$.

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

In Theory: Do not want to use **the same** e over and over again for fear of this being exploited.

Who is Right: $e = 2^{16} + 1$ is used a lot. Should it be?

- Nobody in academia has cracked RSA just using that $e = 2^{2^4} - 1$.
- Nobody in the real world has cracked RSA just using that $e = 2^{2^4} - 1$.

$e = 2^{2^4} + 1$ vs My Fears

In Practice: Want to use $e = 2^{2^4} + 1$ since:

1. Only 15 mults.
2. $2^{2^4} + 1$ Big enough to ward off the low-e attacks (we will study those later).
3. $2^{2^4} + 1$ is prime, so only way it fails to be rel prime to $R = (p-1)(q-1)$. is if it divides R. Unlikely and easily tested.

In Theory: Do not want to use **the same** e over and over again for fear of this being exploited.

Who is Right: $e = 2^{16} + 1$ is used a lot. Should it be?

- Nobody in academia has cracked RSA just using that $e = 2^{2^4} - 1$.
- Nobody in the real world has cracked RSA just using that $e = 2^{2^4} - 1$.
- Do we really know that?

RSA has NY,NY Problem. Will Fix

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

What can Eve **easily** deduce?

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

What can Eve **easily** deduce?

Eve can know if $c_1 = c_2$ or not. So what?

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

What can Eve **easily** deduce?

Eve can know if $c_1 = c_2$ or not. So what?
Eve knows if $m_1 = m_2$ or not. Its the NY,NY problem!

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

What can Eve **easily** deduce?

Eve can know if $c_1 = c_2$ or not. So what?
Eve knows if $m_1 = m_2$ or not. Its the NY,NY problem!

That alone makes it insecure.

Plain RSA Bytes!

The RSA given above is referred to as **Plain RSA**.
Insecure!

Scenario
Eve sees Bob send Alice c_1 (message is m_1).
Later Eve sees Bob send Alice c_2 (message is m_2).

What can Eve **easily** deduce?

Eve can know if $c_1 = c_2$ or not. So what?
Eve knows if $m_1 = m_2$ or not. Its the NY,NY problem!

That alone makes it insecure.
Plain RSA is never used and should never be used!

PKCS-1.5 RSA

How can we fix RSA to make it work? **Discuss**

PKCS-1.5 RSA

How can we fix RSA to make it work? **Discuss** Need randomness.

PKCS-1.5 RSA

How can we fix RSA to make it work? **Discuss** Need randomness.

We need to change how Bob sends a message;
BAD To send $m \in \{1, \ldots, N-1\}$, send $m^e \pmod{N}$.

PKCS-1.5 RSA

How can we fix RSA to make it work? **Discuss** Need randomness.

We need to change how Bob sends a message;
BAD To send $m \in \{1, \ldots, N-1\}$, send $m^e \pmod{N}$.

FIX To send $m \in \{1, \ldots, N-1\}$, pick rand r, send $(rm)^e$.
(NOTE- rm means r CONCAT with m here and elsewhere.) Alice and Bob agree on **length** of r ahead of time.

PKCS-1.5 RSA

How can we fix RSA to make it work? **Discuss** Need randomness.

We need to change how Bob sends a message;
BAD To send $m \in \{1, \ldots, N-1\}$, send $m^e \pmod{N}$.

FIX To send $m \in \{1, \ldots, N-1\}$, pick rand r, send $(rm)^e$.
(NOTE- rm means r CONCAT with m here and elsewhere.) Alice and Bob agree on **length** of r ahead of time.

Alice and Bob pick L_1 and L_2 such that $\lg N = L_1 + L_2$.
To send $m \in \{0,1\}^{L_2}$ pick random $r \in \{0,1\}^{L_1}$.
When Alice gets rm she will know that m is the last L_2 bits.

Example

$p = 31,\ q = 37,\ N = pq = 31 \times 37 = 1147.$

Example

$p = 31,\ q = 37,\ N = pq = 31 \times 37 = 1147.$
$R = \phi(N) = 30 * 36 = 1080$

Example

$p = 31,\ q = 37,\ N = pq = 31 \times 37 = 1147.$
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3.$

Example

$p = 31$, $q = 37$, $N = pq = 31 \times 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3$.
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

Example

$p = 31,\ q = 37,\ N = pq = 31 \times 37 = 1147.$
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3.$
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

1. Bob generates $L_1 = 3$ random bits. 100.

Example

$p = 31$, $q = 37$, $N = pq = 31 \times 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3$.
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

1. Bob generates $L_1 = 3$ random bits. 100.
2. Bob sends 1001100100 which is 612 in base 10 by sending $612^{77} \pmod{1147}$ which is 277.

Example

$p = 31$, $q = 37$, $N = pq = 31 \times 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3$.
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

1. Bob generates $L_1 = 3$ random bits. 100.
2. Bob sends 1001100100 which is 612 in base 10 by sending $612^{77} \pmod{1147}$ which is 277.
3. Alice decodes by doing $277^{533} \pmod{1147} = 612$.

Example

$p = 31$, $q = 37$, $N = pq = 31 \times 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3$.
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

1. Bob generates $L_1 = 3$ random bits. 100.
2. Bob sends 1001100100 which is 612 in base 10 by sending $612^{77} \pmod{1147}$ which is 277.
3. Alice decodes by doing $277^{533} \pmod{1147} = 612$.
4. Alice puts 612 into binary to get 1001100100. She knows to only read the last 7 bits 1100100.

Example

$p = 31$, $q = 37$, $N = pq = 31 \times 37 = 1147$.
$R = \phi(N) = 30 * 36 = 1080$
$e = 77$ (e rel prime to R), $d = 533$ ($ed \equiv 1 \pmod{R}$).
$L_1 = 3$.
Bob wants to send 1100100 (note- $L_2 = 7$ bits).

1. Bob generates $L_1 = 3$ random bits. 100.
2. Bob sends 1001100100 which is 612 in base 10 by sending $612^{77} \pmod{1147}$ which is 277.
3. Alice decodes by doing $277^{533} \pmod{1147} = 612$.
4. Alice puts 612 into binary to get 1001100100. She knows to only read the last 7 bits 1100100.

Important If later Bob wants to send 100100 again he will choose a DIFFERENT random 3 bits so Eve won't know he sent the same message.

RSA has Another Problem

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.
What can Eve do that is still obnoxious?

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.
What can Eve do that is still obnoxious?
Eve can compute $2^e(rm)^e \equiv (2(rm))^e \pmod{N}$. So what?

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.
What can Eve do that is still obnoxious?
Eve can compute $2^e(rm)^e \equiv (2(rm))^e \pmod{N}$. So what?

Eve can later pretend she is Bob and send $(2(rm))^e \pmod{N}$.

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.
What can Eve do that is still obnoxious?
Eve can compute $2^e(rm)^e \equiv (2(rm))^e \pmod{N}$. So what?

Eve can later pretend she is Bob and send $(2(rm))^e \pmod{N}$.

Why bad? **Discuss**

Is PKCS-1.5 RSA Secure?

Is PKCS-1.5 RSA Secure? VOTE

- YES (under hardness assumptions and large n)
- NO (there is yet another weird security thing we overlooked)

NO (there is yet another weird security thing we overlooked)
Scenario N and e are public. Bob sends $(rm)^e \pmod{N}$.
Eve cannot determine what m is.
What can Eve do that is still obnoxious?
Eve can compute $2^e(rm)^e \equiv (2(rm))^e \pmod{N}$. So what?

Eve can later pretend she is Bob and send $(2(rm))^e \pmod{N}$.

Why bad? **Discuss**
(1) will confuse Alice (2) Sealed Bid Scenario.

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

1. The definition above is informal.

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

1. The definition above is informal.
2. Can modify RSA so that it's probably not malleable.

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

1. The definition above is informal.
2. Can modify RSA so that it's probably not malleable.
3. That way is called PKCS-2.0-RSA.

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

1. The definition above is informal.
2. Can modify RSA so that it's probably not malleable.
3. That way is called PKCS-2.0-RSA.
4. Name BLAH-1.5 is hint that it's not final version.

Malleability

An encryption system is **malleable** if when Eve sees a message she can figure out a way to send a similar one, where she knows the similarity (she still does not know the message).

1. The definition above is informal.
2. Can modify RSA so that it's probably not malleable.
3. That way is called PKCS-2.0-RSA.
4. Name BLAH-1.5 is hint that it's not final version.
5. There are other issues that RSA needs to deal with and does, so the real RSA that is used adds more to what I've said here.

Other Public Key Systems

Better Hardness Assumptions

We really want to say

Cracking RSA is Exactly as Hard as Factoring

but we do not know this, and it's probably false.

Better Hardness Assumptions

We really want to say
Cracking RSA is Exactly as Hard as Factoring
but we do not know this, and it's probably false.

Are there other Public Key Cryptosystems that **are** equivalent to factoring?

Better Hardness Assumptions

We really want to say
Cracking RSA is Exactly as Hard as Factoring
but we do not know this, and it's probably false.

Are there other Public Key Cryptosystems that **are** equivalent to factoring?

Yes. On Next Slide.

Rabin's Encryption System and its Variants

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.
5. Cracking Rabin Enc EQUIV factoring: but this is only if Eve has no other information.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.
5. Cracking Rabin Enc EQUIV factoring: but this is only if Eve has no other information.
6. If Eve can trick Alice into sending a chosen message, she can crack Rabin. So **Chosen Plaintext Attack**-insecure.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.
5. Cracking Rabin Enc EQUIV factoring: but this is only if Eve has no other information.
6. If Eve can trick Alice into sending a chosen message, she can crack Rabin. So **Chosen Plaintext Attack**-insecure.

Why is RSA used and not Rabin? either

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.
5. Cracking Rabin Enc EQUIV factoring: but this is only if Eve has no other information.
6. If Eve can trick Alice into sending a chosen message, she can crack Rabin. So **Chosen Plaintext Attack**-insecure.

Why is RSA used and not Rabin? either

1. The problems above make it not practical.

Rabin's Encryption System and its Variants

1. Rabin's enc equivalent to factoring pq.
2. Rabin's enc is hard to use: messages do not decode uniquely.
3. Blum-Williams modified Rabin's Enc so that messages decode uniquely; but the set of messages you can send is small.
4. Hard to combine Blum-Williams modification with the padding needed to solve NY,NY problem.
5. Cracking Rabin Enc EQUIV factoring: but this is only if Eve has no other information.
6. If Eve can trick Alice into sending a chosen message, she can crack Rabin. So **Chosen Plaintext Attack**-insecure.

Why is RSA used and not Rabin? either

1. The problems above make it not practical.
2. The problems above could have been gotten around but RSA just got to the market faster.

RSA Summary

Summary of RSA

Summary of RSA

1. PKCS-2.0-RSA is REALLY used!

Summary of RSA

1. PKCS-2.0-RSA is REALLY used!
2. There are many variants of RSA but all use the ideas above.

Summary of RSA

1. PKCS-2.0-RSA is REALLY used!
2. There are many variants of RSA but all use the ideas above.
3. Factoring easy implies RSA crackable. TRUE.

Summary of RSA

1. PKCS-2.0-RSA is REALLY used!
2. There are many variants of RSA but all use the ideas above.
3. Factoring easy implies RSA crackable. TRUE.
4. RSA crackable implies Factoring easy: UNKNOWN.

Summary of RSA

1. PKCS-2.0-RSA is REALLY used!
2. There are many variants of RSA but all use the ideas above.
3. Factoring easy implies RSA crackable. TRUE.
4. RSA crackable implies Factoring easy: UNKNOWN.
5. RSA crackable implies Factoring easy: Often stated in expositions of crypto. They are wrong!

How Important Is Public Key?

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards
2. Ebay – Paypal

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards
2. Ebay – Paypal
3. Facebook privacy –

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards
2. Ebay – Paypal
3. Facebook privacy – just kidding, Facebook has no privacy. see: `https://www.youtube.com/watch?v=cqggW08BWO0`

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards
2. Ebay – Paypal
3. Facebook privacy – just kidding, Facebook has no privacy. see: `https://www.youtube.com/watch?v=cqggW08BWO0`
4. Every financial institution in the world.

Used Everywhere

Public key is mostly used for giving out keys to be used for classical systems.
This makes the following work:

1. Amazon – Credit Cards
2. Ebay – Paypal
3. Facebook privacy – just kidding, Facebook has no privacy. see: `https://www.youtube.com/watch?v=cqggW08BWO0`
4. Every financial institution in the world.
5. Military – though less is publicly known about this.

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:
    1.1 Math-advances have sped up factoring by 1000 times.

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:
   1.1 Math-advances have sped up factoring by 1000 times.
   1.2 Hardware-advances have sped up factoring by 1000 times.

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:
    1.1 Math-advances have sped up factoring by 1000 times.
    1.2 Hardware-advances have sped up factoring by 1000 times.
    1.3 So Factoring has been sped up 1,000,000 times.

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:
   1.1 Math-advances have sped up factoring by 1000 times.
   1.2 Hardware-advances have sped up factoring by 1000 times.
   1.3 So Factoring has been sped up 1,000,000 times.
2. Factoring is in Quantum P, though making that practical seems a ways off.

Public Key Not Based on Factoring

What if Factoring can be done fast (quantum, fancy number theory, better hardware)?

1. Since 1960:
   1.1 Math-advances have sped up factoring by 1000 times.
   1.2 Hardware-advances have sped up factoring by 1000 times.
   1.3 So Factoring has been sped up 1,000,000 times.
2. Factoring is in Quantum P, though making that practical seems a ways off.
3. There are now several Public Key Systems based on **other** hardness assumptions. See next slide.

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.
2. **Lattice-based Crypto** Based on certain lattice problems being hard to solve. Has been around since 1995.

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.
2. **Lattice-based Crypto** Based on certain lattice problems being hard to solve. Has been around since 1995.
3. **Learning-With Errors (LWE)** Based on the difficulty of learning a function from just a few points. Has been around since 2000. We will cover this later.

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.
2. **Lattice-based Crypto** Based on certain lattice problems being hard to solve. Has been around since 1995.
3. **Learning-With Errors (LWE)** Based on the difficulty of learning a function from just a few points. Has been around since 2000. We will cover this later.
4. **McElice Public Key** Based on error-correcting codes. Hardness assumption is that its hard to error-correct without the parity matrix. Has been around since 1978 but large keys made it a problem.

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.
2. **Lattice-based Crypto** Based on certain lattice problems being hard to solve. Has been around since 1995.
3. **Learning-With Errors (LWE)** Based on the difficulty of learning a function from just a few points. Has been around since 2000. We will cover this later.
4. **McElice Public Key** Based on error-correcting codes. Hardness assumption is that its hard to error-correct without the parity matrix. Has been around since 1978 but large keys made it a problem.

None of these are widely used

Public Key Not Based on Factoring (cont)

Non-factoring based crypto systems:

1. **Elliptic Curve Cryto** Based on elliptic curves (duh). Classically this is better than RSA since is secure with smaller parameters. However, a quantum computer can crack it. Has been around since 1985 but hard math made it hard to use.
2. **Lattice-based Crypto** Based on certain lattice problems being hard to solve. Has been around since 1995.
3. **Learning-With Errors (LWE)** Based on the difficulty of learning a function from just a few points. Has been around since 2000. We will cover this later.
4. **McElice Public Key** Based on error-correcting codes. Hardness assumption is that its hard to error-correct without the parity matrix. Has been around since 1978 but large keys made it a problem.

None of these are widely used Why?

Why Aren't The NON-RSA Systems Used?

Why Aren't The NON-RSA Systems Used?

1. Chicken-and-egg problem: since they have not been out there and attacked, and fixed (like RSA) they are not considered secure.

Why Aren't The NON-RSA Systems Used?

1. Chicken-and-egg problem: since they have not been out there and attacked, and fixed (like RSA) they are not considered secure.
2. Inertia.

Why Aren't The NON-RSA Systems Used?

1. Chicken-and-egg problem: since they have not been out there and attacked, and fixed (like RSA) they are not considered secure.
2. Inertia.
3. Changing over would be expensive and a company has to ask itself, is it worth it?

Why Aren't The NON-RSA Systems Used?

1. Chicken-and-egg problem: since they have not been out there and attacked, and fixed (like RSA) they are not considered secure.
2. Inertia.
3. Changing over would be expensive and a company has to ask itself, is it worth it?
4. There are other security issues that are more pressing.

Why Aren't The NON-RSA Systems Used?

1. Chicken-and-egg problem: since they have not been out there and attacked, and fixed (like RSA) they are not considered secure.
2. Inertia.
3. Changing over would be expensive and a company has to ask itself, is it worth it?
4. There are other security issues that are more pressing.However, they are also not being dealt with.

Will These Systems be Used?

NIST (National Institute of Standards and Technology) solicited **Quantum-Resistant Crypto Systems**.

Will These Systems be Used?

NIST (National Institute of Standards and Technology) solicited **Quantum-Resistant Crypto Systems**.

Lattice-Based, LWE, and Code based all made it into the 2nd round:

Will These Systems be Used?

NIST (National Institute of Standards and Technology) solicited **Quantum-Resistant Crypto Systems**.

Lattice-Based, LWE, and Code based all made it into the 2nd round:

`https://www.scribd.com/document/474476570/PQC-Overview-Aug-2020-NIST`

BILL, STOP RECORDING LECTURE!!!!

BILL STOP RECORDING LECTURE!!!