Public Key Crypto: RSA
Public Key Cryptography: ElGamal and RSA
Article Title: Whack a Mole: The new president (of Colombia) calls off talks with a lesser-known leftist insurgent group.
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?:

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
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.
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: ElGamal
Recall Diffie-Hellman

1. Alice and Bob end up sharing a secret.
2. $p, g$ are public keys.
3. Under a hardness assumption Eve does not know the secret.
4. The secret is not in Alice or Bob’s control

DH cannot be used for the following:

Alice takes the message *Lets do our Math/CMSC 456 HW on time this week for a change* encrypt it, send it to Bob, and Bob Decrypts it.

We describe the ElGamal Public Key Encryption Scheme where Alice and Bob can encrypt and decrypt under a hardness assumption.
ElGamal is DH with a Twist

1. Alice and Bob do Diffie Hellman.
2. Alice and Bob share secret $s = g^{ab}$.
3. Alice and Bob compute $s^{-1} \pmod{p}$.
4. To send $m$, Alice sends $c = ms \pmod{p}$
5. To decrypt, Bob computes $cs^{-1} \equiv mss^{-1} \equiv m \pmod{p}$

We omit discussion of Hardness assumption (HW)
ElGasarch is DH with a Twist

1. Alice and Bob do Diffie Hellman over mod $p$. Let $n = \lceil \lg p \rceil$. All elements of $\mathbb{Z}_p^*$ are $n$-bit strings.
2. Alice and Bob share secret $s = g^{ab}$. View as a bit string.
3. To send $m$, Alice sends $c = m \oplus s$
4. To decrypt, Bob computes $c \oplus s = c \oplus s \oplus s = m \pmod{p}$

Why is ElGamal used and ElGasarch is not? Discuss
ElGasarch is DH with a Twist

1. Alice and Bob do Diffie Hellman over mod $p$. Let $n = \lceil \lg p \rceil$. All elements of $\mathbb{Z}_p^*$ are $n$-bit strings.
2. Alice and Bob share secret $s = g^{ab}$. View as a bit string.
3. To send $m$, Alice sends $c = m \oplus s$
4. To decrypt, Bob computes $c \oplus s = c \oplus s \oplus s = m \pmod{p}$

Why is ElGamal used and ElGasarch is not? Discuss

Example: $p = 23$. The elements are $\{0, \ldots, 22\}$. 0, \ldots, 15 use 4 bits. 16, \ldots, 22 use 5 bits. So if all use 5 bits then $15/22 \sim 0.68$ of the strings have a 0 as first bit. Not Random Enough.

Could ElGasarch work with some variant of DH? Discuss
ElGasarch is DH with a Twist

1. Alice and Bob do Diffie Hellman over mod \( p \). Let \( n = \lceil \lg p \rceil \). All elements of \( \mathbb{Z}_p^* \) are \( n \)-bit strings.
2. Alice and Bob share secret \( s = g^{ab} \). View as a bit string.
3. To send \( m \), Alice sends \( c = m \oplus s \)
4. To decrypt, Bob computes \( c \oplus s = c \oplus s \oplus s = m \pmod p \)

Why is ElGamal used and ElGasarch is not? Discuss

Example: \( p = 23 \). The elements are \( \{0, \ldots, 22\} \). 0, \ldots, 15 use 4 bits. 16, \ldots, 22 use 5 bits. So if all use 5 bits then \( 15/22 \sim 0.68 \) of the strings have a 0 as first bit. Not Random Enough.

Could ElGasarch work with some variant of DH? Discuss

Would need to do DH over a group (1) with power-of-2 elts, (2) DL is hard, (3) mult is easy. Do any exist? Do not know.
Public Key Cryptography: RSA
Exponentiation Mod $p$ Revisited

**Recall** If $p$ prime, $a \not\equiv 0 \pmod{p}$, then $a^{p-1} \equiv 1 \pmod{p}$.

How to compute $3^{1000} \pmod{7}$? Could do repeated squaring. Can we do better? Discuss.

Yes. By recall with $p=7$ and $a=3$ we have $3^6 \equiv 1 \pmod{7}$.

$3^6 \equiv (3^6)^{k} \equiv 1^k \equiv 1$.

So $3^{1000} \equiv 3^6 \times 166 + 4 \equiv (3^6)^{166} \times 3^4 \equiv 3^4$. 

Exponentiation Mod $p$ Revisited

Recall If $p$ prime, $a \not\equiv 0 \pmod{p}$, then $a^{p-1} \equiv 1 \pmod{p}$.
How to compute $3^{1000} \pmod{7}$?
Could do repeated squaring. Can we do better? Discuss.
Exponentiation Mod $p$ Revisited

Recall If $p$ prime, $a \not\equiv 0 \pmod{p}$, then $a^{p-1} \equiv 1 \pmod{p}$.

How to compute $3^{1000} \pmod{7}$?

Could do repeated squaring. Can we do better? Discuss. Yes

By Recall with $p = 7$ and $a = 3$ we have

$$3^6 \equiv 1 \pmod{7}.$$
Recall If \( p \) prime, \( a \not\equiv 0 \pmod{p} \), then \( a^{p-1} \equiv 1 \pmod{p} \).

How to compute \( 3^{1000} \pmod{7} \)?

Could do repeated squaring. Can we do better? Discuss. Yes

By Recall with \( p = 7 \) and \( a = 3 \) we have

\[
3^6 \equiv 1 \quad \text{(mod } 7\text{)}.
\]

\[
3^{6k} \equiv (3^6)^k \equiv 1^k \equiv 1.
\]
Exponentiation Mod $p$ Revisited

Recall If $p$ prime, $a \not\equiv 0 \pmod{p}$, then $a^{p-1} \equiv 1 \pmod{p}$.

How to compute $3^{1000} \pmod{7}$?

Could do repeated squaring. Can we do better? Discuss. Yes

By Recall with $p = 7$ and $a = 3$ we have

$$3^6 \equiv 1 \pmod{7}.$$ 

$$3^{6k} \equiv (3^6)^k \equiv 1^k \equiv 1.$$ 

so

$$3^{1000} \equiv 3^{6 \times 166 + 4} \equiv (3^6)^{166} \times 3^4 \equiv 3^4$$
Expoentiation Mod $p$ Revisited: Another Example

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

Repeated squaring would take at least $\lg(999,999,999) \sim 30 \times$’s.
Exponentiation Mod $p$ Revisited: Another Example

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

Repeated squaring would take at least $\lg(999,999,999) \sim 30 \times$’s.

$999,999,999 \equiv 27 \pmod{106}$

This took one division, same cost as one $\times$. 
Exponentiation Mod $p$ Revisited: Another Example

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

Repeated squaring would take at least $\log(999,999,999) \sim 30 \times$’s.

$$999,999,999 \equiv 27 \pmod{106}$$

This took one division, same cost as one $\times$.

$$11^{999,999,999} \equiv 11^{27} \pmod{107}$$
Exponentiation Mod $p$ Revisited: Another Example

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

Repeated squaring would take at least $\log(999,999,999) \approx 30 \times$’s.

$999,999,999 \equiv 27 \pmod{106}$

This took one division, same cost as one $\times$.

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

Now do normal repeated squaring. $27 = (11011)_2$. $9 \times$’s. So only $10 \times$’s total.
Exponentiation Mod $p$ Revisited: Another Example

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

Repeated squaring would take at least $\lg(999,999,999) \sim 30 \times'$s.

$999,999,999 \equiv 27 \pmod{106}$

This took one division, same cost as one $\times$.

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

Now do normal repeated squaring. $27 = (11011)_2$. $9 \times'$s. So only $10 \times'$s total.

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

$n = k(p - 1) + r$ where $0 \leq r \leq p - 2$ and $r \equiv n \pmod{p - 1}$.

$$a^n \equiv a^{k(p-1)+r} \equiv (a^{p-1})^k \times a^r \equiv a^n \pmod{p-1} \pmod{p}.$$
Needed Mathematics- The $\phi$ Function

Next few slides are on the $\phi$ function.

YES, you have already seen it.

Who first said
Math is best learned twice... at least twice.
Needed Mathematics- The $\phi$ Function

Next few slides are on the $\phi$ function.

YES, you have already seen it.

Who first said

Math is best learned twice... at least twice.

My CMSC 452 class thought either Gauss or Gasarch.
Needed Mathematics- The $\phi$ Function

Next few slides are on the $\phi$ function.

YES, you have already seen it.

Who first said
Math is best learned twice... at least twice.
My CMSC 452 class thought either Gauss or Gasarch.

Answer: Said by Larry Denenberg, who was a grad student in CS the same time Bill Gasarch was.
Needed Mathematics- The $\phi$ Function

Next few slides are on the $\phi$ function.

YES, you have already seen it.

Who first said
Math is best learned twice... at least twice.
My CMSC 452 class thought either Gauss or Gasarch.

**Answer:** Said by Larry Denenberg, who was a grad student in CS the same time Bill Gasarch was. Popularized by Bill Gasarch.
Next few slides are on the $\phi$ function.

YES, you have already seen it.

Who first said
Math is best learned twice... at least twice.
My CMSC 452 class thought either Gauss or Gasarch.

**Answer:** Said by Larry Denenberg, who was a grad student in CS the same time Bill Gasarch was. Popularized by Bill Gasarch. Probably not said by Gauss.
Next few slides are on the \( \phi \) function.

YES, you have already seen it.

Who first said
Math is best learned twice... at least twice.
My CMSC 452 class thought either Gauss or Gasarch.

Answer: Said by Larry Denenberg, who was a grad student in CS the same time Bill Gasarch was. Popularized by Bill Gasarch. Probably not said by Gauss. Probably not true for Gauss.
Needed Mathematics- The $\phi$ Function

**Recall** If $p$ is prime and $1 \leq a \leq p - 1$ then $a^{p-1} \equiv 1 \pmod{p}$.
Hence
**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 $\phi(ab) = \phi(a)\phi(b)$.
Needed Mathematics - Examples

**Theorem** (Fermat-Euler Theorem) If \(a, n\) rel prime then \(a^{\phi(n)} \equiv 1 \pmod{n}\).

Can use for computation similar to using \(a^{p-1} \equiv 1 \pmod{p}\).

\[
14^{999,999} \pmod{393}
\]

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

\[
14^{999,999} = 14^{3846 \times 260 + 39} \pmod{500} \equiv (14^{260})^{3846} \times 14^{39} \pmod{500}
\]

\[
\equiv 1 \times 14^{39} \pmod{500}
\]
Theorem for Primes, Theorem for $n$

For Primes $p$ prime, $a \not\equiv 0 \pmod{p}$, $n \in \mathbb{N}$.

$n = k(p - 1) + r$ where $0 \leq r \leq p - 2$ and $r \equiv n \pmod{p - 1}$.

\[
a^n \equiv a^{k(p-1)+r} \equiv (a^{p-1})^k \times a^r \equiv a^n \pmod{p-1} \pmod{p}.
\]

For Composites $a, n$ rel prime.

$n = k\phi(n) + r$ where $0 \leq r \leq \phi(n) - 1$ and $r \equiv n \pmod{\phi(n)}$.

\[
a^n \equiv a^{k\phi(n)+r} \equiv (a^{\phi(n)})^k \times a^r \equiv a^n \pmod{\phi(n)} \pmod{n}.
\]
Easy and Hard

**Known to be Easy**

1. Given \( n \), generate two primes of length: \( p, q \), and finding \( N = pq \) and \( R = (p - 1)(q - 1) \).
2. Given \( p, q, R \) as above, finding \( e \) rel prime to \( R \).
3. Given \( p, q, R, e \) as above, finding \( d \) such that \( ed \equiv 1 \pmod{R} \). **KEY:** Easy since have \( p, q \). Would be hard otherwise
4. Given \( N, m, e \) as above, compute \( m^e \pmod{N} \).

**Thought to be Hard**

Given \( N, e \) as above find \( d \) as above. **Note that we are not given \( p, q \) or \( R \).**
RSA

Let $n$ be a security parameter

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

PRO: Alice and Bob can execute the protocol easily.

Biggest PRO: Alice and Bob never had to meet!

Question: Can Eve find out $m$?
RSA

Let $n$ be a security parameter

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

Let $n$ be a security parameter

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

Note: Works for any $1 \leq m \leq p-1$.
$m$ need not be rel prime to $N$.

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

Question: Can Eve find out $m$?
RSA

Let $n$ be a security parameter

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

Note: Works for any $1 \leq m \leq p - 1$. $m$ need not be rel prime to $N$.

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

Question: Can Eve find out $m$?
RSA

Let $n$ be a security parameter

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

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

Note: Works for any $1 \leq m \leq p - 1$.

m need not be rel prime to N.

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

Question: Can Eve find out $m$?
RSA

Let \( n \) be a security parameter

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

\[
(m^e)^d \equiv m^{ed} \equiv m^{ed} \pmod{R} \equiv m^1 \pmod{R} \equiv m
\]

Note: Works for any \( 1 \leq m \leq p - 1 \). \( m \) need not be rel prime to \( N \).
Let $n$ be a security parameter

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

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

Note: Works for any $1 \leq m \leq p - 1$. $m$ need not be rel prime to $N$.

PRO: Alice and Bob can execute the protocol easily.
RSA

Let $n$ be a security parameter

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

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

   Note: Works for any $1 \leq m \leq p - 1$. $m$ need not be rel prime to $N$.

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

Let $n$ be a security parameter

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

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

Note: Works for any $1 \leq m \leq p - 1$. $m$ need not be rel prime to $N$.

PRO: Alice and Bob can execute the protocol easily.

Biggest PRO: Alice and Bob never had to meet!

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.
Do RSA in Class

Pick out two students to be Alice and Bob.

Use primes

\[ p = 31, \text{ Prime} \]
\[ q = 37, \text{ Prime} \]

\[ N = pq = 31 \times 37 = 1147. \]
\[ R = \phi(N) = 30 \times 36 = 1080 \]

\[ e = 77, \text{ } e \text{ rel prime to } R \]
\[ d = 533 \text{ (} ed \equiv 1 \text{ (mod } R)) \]

CHECK: \[ ed = 77 \times 533 = 41041 \equiv 1 \text{ (mod } 1080) \].

Bob: pick an \( m \in \{1, \ldots, N - 1\} = \{1, \ldots, 1147\} \). Do not tell us what it is.

Bob: compute \( c = m^e \text{ (mod } 1147) \) and tell it to us.

Alice: compute \( c^d \text{ (mod } 1147) \), should get back \( m \).
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.
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.

Note: In ugrad math classes rare to have a statement that is
UNKNOWN TO SCIENCE. Discuss.
**Hardness Assumption**

**Definition**
Let $f$ be the following function:

**Input:** $N, e, m^e \pmod{N}$ (known $N = pq$ but not known what $p, q$ are).

**Outputs:** $m$.

**Hardness assumption (HA):** $f$ 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 that Eve is not obliged to work in.
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
2) Factoring is hard, HA is false. RSA crackable, though Factoring hard!!
3) Factoring is hard, HA is true, but RSA is crackable by other means. Timing Attacks. Must rethink our model of security.
4) Factoring is hard, HA is true, and RSA remains uncracked for years. Increases our confidence but...

Item 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?
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
2) Factoring is hard, HA is false. RSA crackable, though Factoring hard!!

Item 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?
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
2) Factoring is hard, HA is false. RSA crackable, though Factoring hard!!
3) Factoring is hard, HA is true, but RSA is crackable by other means. Timing Attacks. Must rethink our model of security.
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
2) Factoring is hard, HA is false. RSA crackable, though Factoring hard!!
3) Factoring is hard, HA is true, but RSA is crackable by other means. Timing Attacks. Must rethink our model of security.
4) Factoring is hard, HA is true, and RSA remains uncracked for years. Increases our confidence but ....
What Could be True?

The following are all possible:

1) Factoring is easy. Then RSA is crackable.
2) Factoring is hard, HA is false. RSA crackable, though Factoring hard!!
3) Factoring is hard, HA is true, but RSA is crackable by other means. Timing Attacks. Must rethink our model of security.
4) Factoring is hard, HA is true, and RSA remains uncracked for years. Increases our confidence but ....

Item 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?
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.
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.

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.

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 $m^e \pmod{N}$.

GOOD?: To send $m \in \{1, \ldots, N - 1\}$, pick rand $r$, send $(rm)^e$.

(Note- $rm$ means $r$ CONCAT with $m$ here and elsewhere.)
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 \( m^e \) (mod \( N \)).

GOOD?: To send \( m \in \{1, \ldots, N - 1\} \), pick rand \( r \), send \( (rm)^e \).
(NOTE- \( rm \) means \( r \) CONCAT with \( m \) here and elsewhere.)

DEC: Alice can find \( rm \) but doesn’t know divider. How to fix?
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 $m^e \pmod{N}$.

GOOD?: To send $m \in \{1, \ldots, N - 1\}$, pick rand $r$, send $(rm)^e$.
(NOTE- $rm$ means $r$ CONCAT with $m$ here and elsewhere.)

DEC: Alice can find $rm$ but doesn’t know divider. How to fix?
Alice and Bob agree on dividers ahead of time. Agree on $L_1 = \left\lfloor \frac{\lg N}{3} \right\rfloor$, $L_2 = \lfloor \lg N \rfloor - L_1$.
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**

\[ p = 31, \text{ Prime } q = 37, \text{ Prime } N = pq = 31 \times 37 = 1147. \]

\[ R = \phi(N) = 30 \times 36 = 1080 \]

\[ e = 77 \text{ (e rel prime to } R), \text{ d } = 533 \text{ (ed } \equiv 1 \text{ (mod } R)) \]

\[ L_1 = \left\lfloor \frac{\log N}{3} \right\rfloor = 3, \text{ } L_2 = \lfloor \log N \rfloor - L = 7. \]

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 \( 612^{77} \text{ (mod 1147)} \) which is 277.

3. Alice decodes by doing \( 277^{533} \text{ (mod 1147)} = 612 \)

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 100 again he will choose a DIFFERENT random 3 bits so Eve won’t know he sent the same message.
Is PKCS-1.5 RSA Secure? VOTE

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

Eve cannot determine what $m$ is. What can Eve do that is still obnoxious? Eve can compute $2^e(r^m)^e \equiv (2(r^m))^e \pmod{N}$. So what? Eve can later pretend she is Bob and send $(2(r^m))^e \pmod{N}$. Why bad? Discuss (1) will confuse Alice (2) Sealed Bid Scenario.
Is PKCS-1.5 RSA Secure?

- 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?

- YES (under hardness assumptions and large $n$)
- 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.

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.
Is PKCS-1.5 RSA Secure?

- **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?

- 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? 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?

- 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).

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.
Final Points About 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!
6. Timing attacks on RSA bypass the math.