Modular arithmetic and trap door ciphers

Joshua Holden

Rose-Hulman Institute of Technology
http://www.rose-hulman.edu/~holden

RSA Setup

Ronald Rivest, Adi Shamir, Leonard Adleman, 1977.

- Pick two primes p and q.
- Compute $n = pq$.
- Pick *encryption exponent* e such that e and $(p-1)(q-1)$ don't have any common prime factors.
- Make n and e public. Keep p and q private.

RSA Setup: Example

- $p = 53$
- $q = 71$
- $n = pq = 3763$
- $(p-1)(q-1) = 3640 = 2^3 \cdot 5 \cdot 7 \cdot 13$
- $e = 27 = 3^3$
- e and $(p-1)(q-1)$ don't have any common prime factors

RSA Setup: PGP public key block

```
From holden@math.duke.edu  Thu Feb  8 14:07:19 2001
Date: Thu, 8 Feb 2001 14:07:18 -0500
X-Authentication-Warning: hamburg.math.duke.edu: holden set sender to holden@hamburg.math.duke.edu
From: Joshua Holden To: holden@math.duke.edu
Subject: message with PGP block

Here is my PGP block: now you can send me messages!

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.2
Comment: Processed by Mailcrypt 3.5.5, an Emacs/PGP interface

mQCNAznRHaMAAAEEAPix/FD/jF/ixMvd9aIjhZ/K6o2kv/TaGAVkeIG5VZ48jzIa
NTqX1EKDw6aABUiQApqavOaQuiLbi/Ez9HXX9LfcTdcp8u94BKGgmEy6Jv1za08I
2YVL1kUJso6lauryr3Sc8wiQTwx3imohM4ai/1dVuq4Qp2WCBSRdyaafdchdAAUR
tC9Kb3NodWEgSG9sZGVuICgxMDI0IGJpdCkgPGhvbGRlbkBtYXRoLmR1a2UuZWR1
Pg==
=VgE9
-----END PGP PUBLIC KEY BLOCK-----
```

Modular Arithmetic

Karl Friedrich Gauss, 1801.

- Modular Arithmetic = “Wrap-around” computations

Example: Start at 12 o’clock. 5 hours plus 8 hours equals 1 o’clock.

$$5 + 8 \equiv 1 \pmod{12}$$

Example: Start at 12 o’clock. 11 hours times 5 equals 7 o’clock.

$$11 \cdot 5 \equiv 7 \pmod{12}$$

RSA Encryption

Anyone can encrypt, because n and e are public.

- To encrypt, convert your message into a set of *plaintext* numbers P, each less than n.
- For each P, compute $C \equiv P^e \pmod{n}$.
- The numbers C are your *ciphertext*.

RSA Encryption: Example

Send the message “cats and dogs”:

- ca ts an dd og sx
- 0200 1918 0013 0303 1406 1823
- $200^e \equiv 12 \pmod{n}$
- $1918^e \equiv 1918 \pmod{n}$
- $13^e \equiv 1550 \pmod{n}$
- $303^e \equiv 3483 \pmod{n}$
- $1406^e \equiv 2042 \pmod{n}$
- $1823^e \equiv 2735 \pmod{n}$

RSA Encryption: PGP message

```
From holden@math.duke.edu  Thu Feb  8 14:09:25 2001
Date: Thu, 8 Feb 2001 14:09:24 -0500
X-Authentication-Warning: hamburg.math.duke.edu: holden set sender to holden@hamburg.math.duke.edu
From: Joshua Holden To: holden@math.duke.edu
Subject: This message is encrypted

-----BEGIN PGP MESSAGE-----
Version: 2.6.2
Comment: Processed by Mailcrypt 3.5.5, an Emacs/PGP interface

hIwDJF3Jpp91yF0BBAC6gnKTMhGWg9hUELd7WfJgUn7OqObCNmvm9V8ff+tyq0re
nSQqCYw784CAkm5gaUtJ0AW4go2pDyI2hm5ocoVfMeBOJpKeckSchncV9zHSH2zx
jBM8W0NYPAaa7AHFisz19rqxkkt1aQ4W49i7LUxq6rXheoSPMMcHbHyBa/mQEaYA
AABEmtEXwkUSMOh+x4dSM/6ZUswVZznmei9TOw+md8OM+LiOSakw91GT431tJPAN
c44q+q2Yq8ehykaz0sV4fXscPy2H9A0=
=v1z0
-----END PGP MESSAGE-----
```

Trap Door

Leonhard Euler, 1736.

- Let $\phi(n)$ be the number of positive integers less than or equal to n which don't have any common factors with n.

Example: If $n = 15$, then the positive integers less than or equal to n which don't have any common factors with n are 1, 2, 4, 7, 8, 11, 13, 14. So $\phi(15) = 8$.

Trap Door: RSA

In the RSA system $n = pq$, so $\phi(n)$ is the number of positive integers less than or equal to n which don't have p or q as a factor.

- How many positive integers less than or equal to n do have p as a factor? $p, 2p, 3p, \ldots, n = qp$ so there are q of them.
- Similarly, there are p positive integers less than or equal to n with q as a factor.
- Only one positive integer less than or equal to n has both p and q as factors, namely $n = pq$. So we should only count this once.

Trap Door: Formula

- Therefore,

$$\phi(n) = n - p - q + 1 = pq - p - q + 1 = (p-1)(q-1).$$

- This is private! You can't calculate it without knowing p and q.
- Why is this useful?

Euler's Theorem

Euler's Theorem: If x is an integer which has no common prime factors with n, then

$$x^{\phi(n)} \equiv 1 \pmod{n}.$$

- Why is Euler's Theorem true?
- Two versions of the answer: Number Theory and Group Theory

Number Theory idea: We consider the positive integers less than or equal to n which don't have any common factors with n, and multiply each of them by x modulo n. Compare them to the same integers without multiplying by x.

Euler's Theorem: Example (I)

- For $n = 15$, consider

$$x, 2x, 4x, 7x, 8x, 11x, 13x, 14x \pmod{15},$$

and compare them to $1, 2, 4, 7, 8, 11, 13, 14$.

- If we multiply all of the first set we get

$$x^8 \cdot 1 \cdot 2 \cdot 4 \cdot 7 \cdot 8 \cdot 11 \cdot 13 \cdot 14 \pmod{15}$$

and if we multiply all of the second set we get

$$1 \cdot 2 \cdot 4 \cdot 7 \cdot 8 \cdot 11 \cdot 13 \cdot 14 \pmod{15}.$$

- What if we do all of this for $x = 11$?

Euler's Theorem: Example (II)

The first set will be:

- $1 \cdot 11 \equiv 11 \pmod{15}$
- $2 \cdot 11 \equiv 7 \pmod{15}$
- $4 \cdot 11 \equiv 14 \pmod{15}$
- $7 \cdot 11 \equiv 2 \pmod{15}$
- $8 \cdot 11 \equiv 13 \pmod{15}$
- $11 \cdot 11 \equiv 1 \pmod{15}$
- $13 \cdot 11 \equiv 8 \pmod{15}$
- $14 \cdot 11 \equiv 4 \pmod{15}$

Euler's Theorem: Example (III)

- The first set is the same as the second set, only in a different order!
- In fact, this always happens.
- So

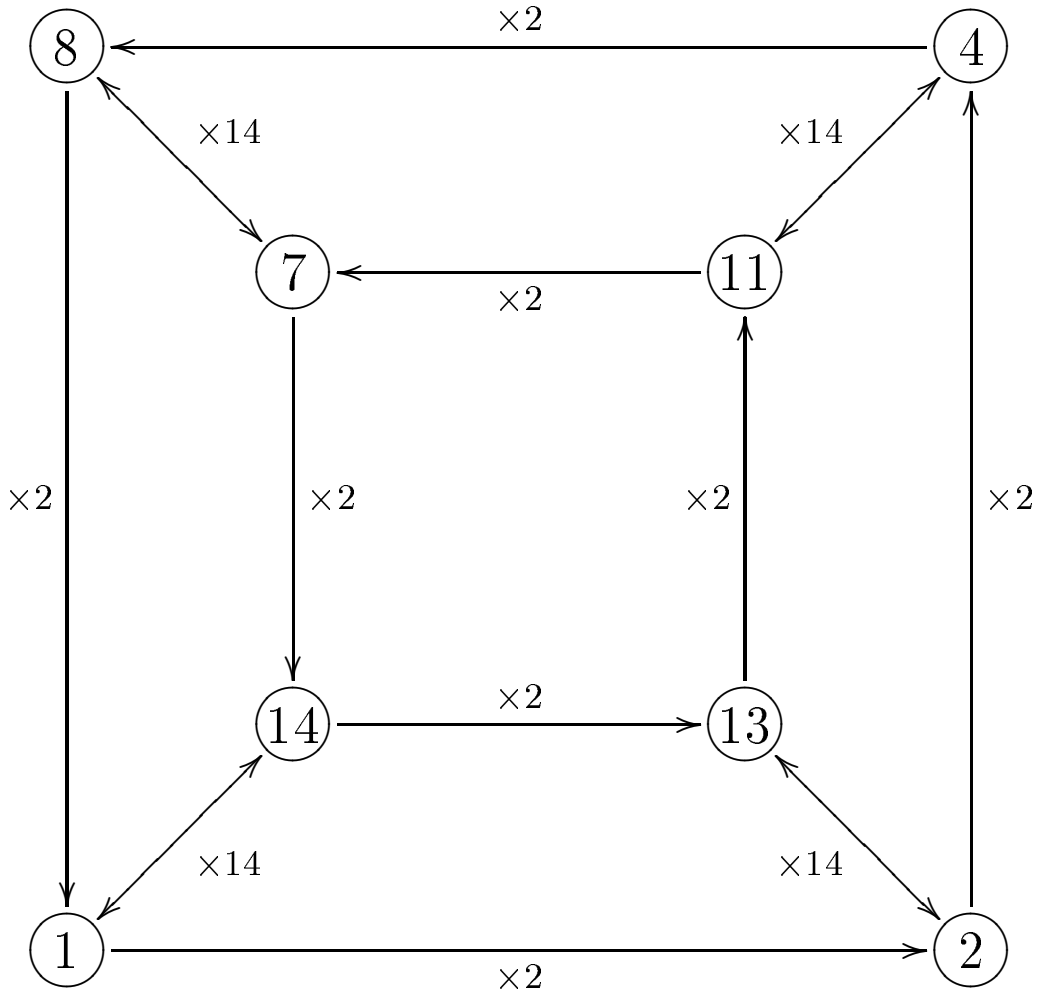
$$x^8 \cdot 1 \cdot 2 \cdot 4 \cdot 7 \cdot 8 \cdot 11 \cdot 13 \cdot 14 \equiv 1 \cdot 2 \cdot 4 \cdot 7 \cdot 8 \cdot 11 \cdot 13 \cdot 14 \pmod{15}$$

or

$$x^8 \equiv 1 \pmod{15}.$$

Cayley diagram

Arthur Cayley, 1878.



Group Theory idea: We make a *Cayley diagram* for the numbers less than n.

Cayley diagram: Example (II)

- Say $x = 11$. Follow the arrows from 1 to 11. This is one x14 arrow and two x2 arrows. If you do this 7 more times, you will be following a total of eight x14 arrows and sixteen x2 arrows, and you should end up at 11 to the eighth. However, eight x14 arrows and sixteen x2 arrows clearly ends you up back where you started! (Note that it doesn't matter in what order you follow the arrows....)

So how do we use Euler's Theorem as a trap door?

RSA: One More Piece

- We need one more piece of (private) information, and an ancient Greek mathematician will tell us how to get it.

Euclid, about 300 B.C.E.

Theorem: If a and b don't have any common prime factors, then there are integers c and d such that

$$ac + bd = 1.$$

Euclidean Algorithm

- Since we picked e such that e and $(p-1)(q-1)$ don't have any common prime factors, then there are integers c and d such that

$$(p-1)(q-1)c + ed = 1$$

or

$$\phi(n)c + ed = 1.$$

- Euclid also tells us how to find c and d, using the *Euclidean Algorithm.*
- Once we have found the *decryption exponent d*, which is private, we can decrypt.

RSA Decryption

For each C, compute $C^d \pmod{n}$.

- What will this give you?
- We know $C \equiv P^e \pmod{n}$, although we don't yet know what P is. So

$$C^d \equiv (P^e)^d \equiv P^{ed} \equiv P^{1-\phi(n)c} \equiv P(P^{\phi(n)})^{-c} \pmod{n}.$$

- But $P^{\phi(n)} \equiv 1 \pmod{n}$ by Euler's Theorem!
- So $C^d \equiv P \pmod{n}$ and we get our original plaintext back.

RSA Decryption: Example (I)

- $p = 53$
- $q = 71$
- $(p-1)(q-1) = 3640$
- $e = 27$
- The Euclidean Algorithm tells us

$$16(p-1)(q-1) - 2157e = 1.$$

- $d = -2157$

RSA Decryption: Example (II)

- $12^d \equiv 200 \pmod{n}$
- $1918^d \equiv 1918 \pmod{n}$
- $1550^d \equiv 13 \pmod{n}$
- $3483^d \equiv 303 \pmod{n}$
- $2042^d \equiv 1406 \pmod{n}$
- $2735^d \equiv 1823 \pmod{n}$
- 0200 1918 0013 0303 1406 1823
- ca ts an dd og sx

“cats and dogs”

Breaking RSA: Factoring

So why do we think RSA is secure?

- As far as we know, the only way to break RSA is to find p and q by factoring n. The fastest known factoring algorithm takes something about like

$$e^{(\log n)^{1/3}(\log(\log n))^{2/3}}$$

time units to factor n. (The size of the time unit depends on things like how fast the computer is!)

Breaking RSA: Fast computers

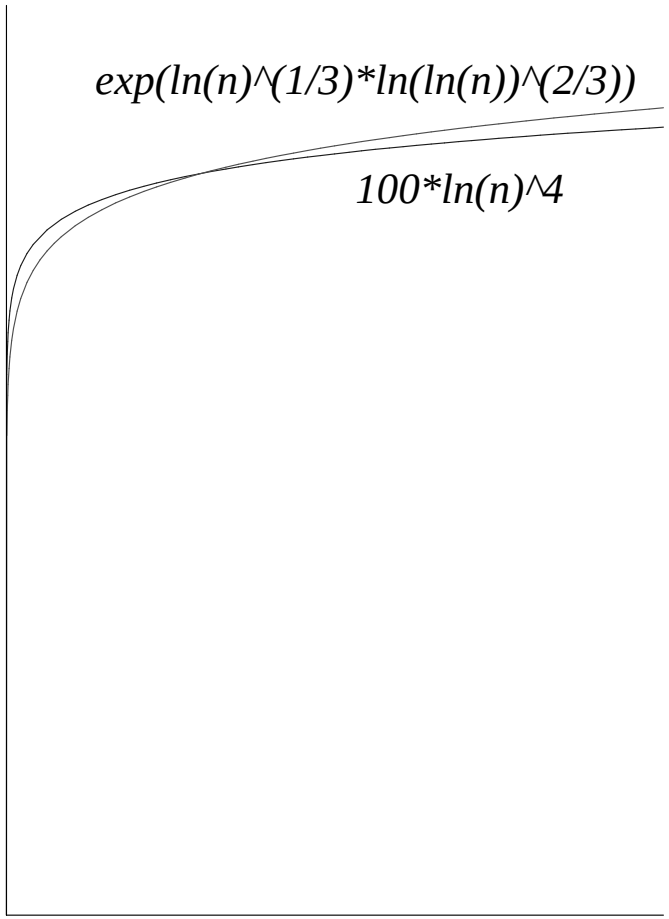
For the fastest single computer in 2006, it would probably take about 1 billion years to factor a number with 300 decimal digits. However, with networked computers, a large company might be able to improve this by a factor of as much as 1 million.

(More technically, it is estimated that factoring a number with 300 decimal digits would take about 1011 MIPS-years. 1 MIPS-year is a million-instructions-per-second processor running for one year. A 1-GHz Pentium is about a 250-MIPS machine.)

Breaking RSA: Factoring vs. Setup

On the other hand, finding p and q and multiplying them together is very fast. Finding a number p which is (probably) prime takes about $100(\log p)^4$ time units. This looks large, but it isn't really; for a 300-digit number this should only take a few minutes. (Multiplying p and q together is even faster.)

Breaking RSA: A Graph



At *some* size of n it will always be easier to make the cipher than to break it!

RSA Digital Signatures

As a bonus, RSA gives us a way to digitally "sign" messages, thereby proving who wrote them. This uses the same public n and e and private d as before.

- For each plaintext P, compute $S \equiv P^d \pmod{n}$.
- The numbers S are your signed message.

RSA Digital Signatures: Example

Sign the message “cats and dogs”:

- ca ts an dd og sx
- 0200 1918 0013 0303 1406 1823
- $200^d \equiv 648 \pmod{n}$
- $1918^d \equiv 1918 \pmod{n}$
- $13^d \equiv 914 \pmod{n}$
- $303^d \equiv 1946 \pmod{n}$
- $1406^d \equiv 664 \pmod{n}$
- $1823^d \equiv 2735 \pmod{n}$

RSA Digital Signatures: PGP message

```
From holden@math.duke.edu  Thu Feb  8 14:10:42 2001
Date: Thu, 8 Feb 2001 14:10:41 -0500
X-Authentication-Warning: hamburg.math.duke.edu: holden set sender to holden@hamburg.math.duke.edu
From: Joshua Holden To: holden@math.duke.edu
Subject: This message is signed but not encrypted

-----BEGIN PGP SIGNED MESSAGE-----

I'm signing this message so that you know it's me!

-----BEGIN PGP SIGNATURE-----
Version: 2.6.2
Comment: Processed by Mailcrypt 3.5.5, an Emacs/PGP interface

iQCVAwUBOoLvKyRdyaafdchdAQELuQP+PBR2lY8rEPrgA4GzWQS/MbE4UDECkgBk
v+6Q/gAwrHzMwemXcZxKU1FGFClvfHxxyjoy8hJgYeLYiGvD+q11gtNGZtTdLzqh
xL/Bdw75fseFxal/32ZS45jMA3gA2220m70hkJg4EzyvlhDUdUI1SIQHn/V26H0g
I25VOm/Ib8s=
=CRW2
-----END PGP SIGNATURE-----
```

Verifying the Signature

Since only you know the decryption exponent d, only you can sign a message. Anyone you send it to can prove it was you by computing $S^e \pmod{n}$ (since n and e are public) and getting back $P^{de} \pmod{n}$, which we know is congruent to P.

- If this matches the P which you sent separately, then the message was correctly signed, so it must have come from someone who knows d.

Verifying the Signature: Example

- $648^e \equiv 200 \pmod{n}$
- $1918^e \equiv 1918 \pmod{n}$
- $914^e \equiv 13 \pmod{n}$
- $1946^e \equiv 303 \pmod{n}$
- $664^e \equiv 1406 \pmod{n}$
- $2735^e \equiv 1823 \pmod{n}$
- 0200 1918 0013 0303 1406 1823
- ca ts an dd og sx

“cats and dogs”

Encrypting and Signing

One can even sign an encrypted message this way. Suppose Alice wants to send Bob an encrypted message.

- She first encrypts with Bob's public n_B and e_B.
- Secondly, she signs the message with her n_A and private d_A. Since her d_A is different from Bob's d_B, they don't cancel out.
- Then Bob can "unsign" the message with Alice's public n_A and e_A.
- Finally, Bob decrypts the message with his n_B and private d_B!

Encrypting and Signing: Example (I)

Alice:

- Private: $p_A = 53, q_A = 71$
- Public: $n_A = p_A q_A = 3763$
- Public: $e_A = 27$
- Private: $d_A = -2157$ (same as before)

Bob:

- Private: $p_B = 41, q_B = 67$
- Public: $n_B = p_B q_B = 2747$
- Private: $(p_B - 1)(q_B - 1) = 2640 = 2^4 \cdot 3 \cdot 5 \cdot 11$
- Public: $e_B = 49 = 7^2$
- Private: The Euclidean Algorithm tells Bob

$$8(p_B - 1)(q_B - 1) - 431e_B = 1.$$

- Private: $d_B - -431$

Encrypting and Signing: Example (II)

Alice encrypts the message with Bob's public information:

- ca ts an dd og sx
- 0200 1918 0013 0303 1406 1823
- $200^{e_B} \equiv 2411 \pmod{n_B}$
- $1918^{e_B} \equiv 1836 \pmod{n_B}$
- $13^{e_B} \equiv 1401 \pmod{n_B}$
- $303^{e_B} \equiv 2314 \pmod{n_B}$
- $1406^{e_B} \equiv 2143 \pmod{n_B}$
- $1823^{e_B} \equiv 1154 \pmod{n_B}$

Encrypting and Signing: Example (III)

Alice signs the message with her private information and send the result to Bob:

- $2411^{d_A} \equiv 2041 \pmod{n_A}$
- $1836^{d_A} \equiv 814 \pmod{n_A}$
- $1401^{d_A} \equiv 1249 \pmod{n_A}$
- $2314^{d_A} \equiv 1396 \pmod{n_A}$
- $2143^{d_A} \equiv 772 \pmod{n_A}$
- $1154^{d_A} \equiv 3139 \pmod{n_A}$

Encrypting and Signing: Example (IV)

Bob "unsigns" the message using Alice's public information:

- $2041^{e_A} \equiv 2411 \pmod{n_A}$
- $814^{e_A} \equiv 1836 \pmod{n_A}$
- $1249^{e_A} \equiv 1401 \pmod{n_A}$
- $1396^{e_A} \equiv 2314 \pmod{n_A}$
- $772^{e_A} \equiv 2143 \pmod{n_A}$
- $3139^{e_A} \equiv 1154 \pmod{n_A}$

Encrypting and Signing: Example (V)

and then decrypts it using his private information:

- $2411^{d_B} \equiv 200 \pmod{n_B}$
- $1836^{d_B} \equiv 1918 \pmod{n_B}$
- $1401^{d_B} \equiv 13 \pmod{n_B}$
- $2314^{d_B} \equiv 303 \pmod{n_B}$
- $2143^{d_B} \equiv 1406 \pmod{n_B}$
- $1154^{d_B} \equiv 1823 \pmod{n_B}$
- 0200 1918 0013 0303 1406 1823
- ca ts an dd og sx

“cats and dogs”

Attacks on RSA

Finding out someone's private d is about as hard as factoring n. But sometimes we can find out a particular message without breaking the general code. Usually this is because e is too small — small e makes the encrypting faster, but can weaken security.

Small Message Attack (I)

- $p = 53$, $q = 71$
- $n = pq = 3763$
- $e = 3$

"abaracadabara"

- ab ar ac ad ab ar ax
- 0001 0017 0002 0003 0002 0017 0023
- $1^e \equiv 1 \pmod{n}$
- $17^e \equiv 1150 \pmod{n}$
- $2^e \equiv 8 \pmod{n}$
- $3^e \equiv 27 \pmod{n}$
- $2^e \equiv 8 \pmod{n}$
- $17^e \equiv 1150 \pmod{n}$
- $23^e \equiv 878 \pmod{n}$

Small Message Attack (II)

But:

- $\sqrt[3]{1} = 1$
- $\sqrt[3]{1150} = 10.4769$
- $\sqrt[3]{8} = 2$
- $\sqrt[3]{27} = 3$
- $\sqrt[3]{8} = 2$
- $\sqrt[3]{1150} = 10.4769$
- $\sqrt[3]{878} = 9.5756$
- 0001 ???? 0002 0003 0002 ???? ????
- ab ?? ac ad ab ?? ??

An eavesdropper can recover most of the message!

Common Exponent Attack (I)

Using a small exponent like $e = 3$ is fast, but it can be insecure. Suppose we're sending the same message to Alice, Bob, and Carol, and they all have the same small exponent.

- $p_A = 53$, $q_A = 71$
- $n_A = p_A q_A = 3763$ (We've used this key before.)
- $e_A = 3$

- $p_B = 41$, $q_B = 83$
- $n_B = p_B q_B = 3403$
- $(p_B - 1)(q_B - 1) = 3280 = 2^4 \cdot 5 \cdot 41$
- $e_B = 3$

- $p_C = 47$, $q_C = 87$
- $n_C = p_C q_C = 4089$
- $(p_C - 1)(q_C - 1) = 3956 = 2^2 \cdot 23 \cdot 43$
- $e_C = 3$

Common Exponent Attack (II)

"cats":

- ca ts
- 0200 1918

Message to Alice:

- $200^{e_A} \equiv 3625 \pmod{n_A}$
- $1918^{e_A} \equiv 2060 \pmod{n_A}$

Message to Bob:

- $200^{e_B} \equiv 2950 \pmod{n_B}$
- $1918^{e_B} \equiv 2223 \pmod{n_B}$

Message to Carol:

- $200^{e_C} \equiv 1916 \pmod{n_C}$
- $1918^{e_C} \equiv 2326 \pmod{n_C}$

Common Exponent Attack (III)

Eve (an eavesdropper) hears the messages. So Eve knows that

$$3625 \equiv P^3 \pmod{n_A}$$
$$2950 \equiv P^3 \pmod{n_B}$$
$$1916 \equiv P^3 \pmod{n_C}$$

and similarly for the second half of the message. (Everything here except P is public information!)

Chinese Remainder Theorem

But:

Chinese Remainder Theorem: If m_1 and m_2 don't have any common prime factors, then

$$x \equiv a \pmod{m_1}, \qquad x \equiv a_2 \pmod{m_2}$$

can be solved for a unique x modulo $m_1 m_2$.

This problem was studied in Greece, China, and India from the first century C.E. on. But the general solution (the *ta-yen*, or "great extension" rule) was first given by Qin Jiushao in 1247.

The *Ta-Yen* Magic Formula

In Eve's case, the *ta-yen* magic formula is:

- $q_A \equiv (n_B n_C)^{-1} \pmod{n_A}$,
- $q_B \equiv (n_A n_C)^{-1} \pmod{n_B}$,
- $q_C \equiv (n_A n_B)^{-1} \pmod{n_C}$,
- $P^3 \equiv 3625 n_B n_C q_A + 2950 n_A n_C q_B + 1916 n_A n_B q_C \pmod{n_A n_B n_C}$
  $\equiv 8000000 \pmod{52361644521}$

The Common Exponent Attack Concluded

But now Eve can use the small message attack:

- $\sqrt[3]{80000000} = 200$
- 0200
- ca (ts)

This is guaranteed to work if there are at least e messages.

First Moral: Small exponents can be dangerous!

Second Moral: Don't send identical messages to different people!

HNAT SOFK LSIR EINT GZXN!