The 2016 Cybersecurity Speaker Series

On the Growth of Cryptography

Ronald L. Rivest, PhD
Professor, Electrical Engineering and Computer Science
Computer Science and Artificial Intelligence Laboratory (CSAIL)
Massachusetts Institute of Technology

Sponsored by:

STRATEGY IS THE BEST SECURITY.
The Executive Master in Cybersecurity
BROWN

On the growth of cryptography[1]

Ronald L. Rivest

Institute Professor
MIT, Cambridge, MA

Cybersecurity Seminar Series
Brown University
April 14, 2016

[1] many slides from my MIT Killian award lecture

Outline

Outline

Euclid – 300 B.C.

There are infinitely many primes:
2, 3, 5, 7, 11, 13, . . .

Euclid – 300 B.C.

There are infinitely many primes:
2, 3, 5, 7, 11, 13, . . .

The greatest common divisor of two numbers is easily computed (using "Euclid's Algorithm"):

$$\gcd(12, 30) = 6$$

Greek Cryptography – The Scytale

An unknown *period* (the circumference of the scytale) is the secret key, shared by sender and receiver.

Pierre de Fermat (1601-1665)
Leonhard Euler (1707–1783)


Fermat's Little Theorem (1640):

For any prime p and any a, $1 \leq a < p$:

$$a^{p-1} = 1 \pmod{p}$$

Pierre de Fermat (1601-1665)
Leonhard Euler (1707–1783)


Fermat's Little Theorem (1640):
For any prime p and any a, $1 \leq a < p$:

$$a^{p-1} = 1 \pmod{p}$$

Euler's Theorem (1736):
If $\gcd(a, n) = 1$, then

$$a^{\phi(n)} = 1 \pmod{n},$$

where $\phi(n) = \#$ of $x < n$ such that $\gcd(x, n) = 1$.

Carl Friedrich Gauss (1777-1855)

Published *Disquisitiones Aritmeticae* at age 21

Carl Friedrich Gauss (1777-1855)

Published *Disquisitiones Aritmeticae* at age 21

“The problem of *distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors* is known to be one of the most important and useful in arithmetic. . . . the dignity of the science itself seems to require solution of a problem so elegant and so celebrated.”

William Stanley Jevons (1835–1882)

Published *The Principles of Science* (1874)

William Stanley Jevons (1835–1882)

Published *The Principles of Science* (1874)

Gave world's first *factoring challenge*:

> *"What two numbers multiplied together will produce 8616460799 ? I think it unlikely that anyone but myself will ever know."*

William Stanley Jevons (1835–1882)

Published *The Principles of Science* (1874)

Gave world's first *factoring challenge*:

> *"What two numbers multiplied together will produce 8616460799 ? I think it unlikely that anyone but myself will ever know."*

Factored by Derrick Lehmer in 1903. (89681 $*$ 96079)

World War I – Radio

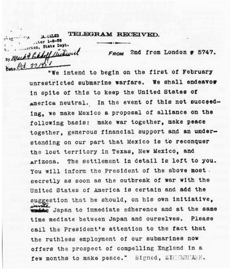
- A marvelous new communication technology—*radio* (Marconi, 1895)—enabled instantaneous communication with remote ships and forces, but also gave all transmitted messages to the enemy.

World War I – Radio

- A marvelous new communication technology—*radio* (Marconi, 1895)—enabled instantaneous communication with remote ships and forces, but also gave all transmitted messages to the enemy.
- Use of cryptography soars.

World War I – Radio

- A marvelous new communication technology—*radio* (Marconi, 1895)—enabled instantaneous communication with remote ships and forces, but also gave all transmitted messages to the enemy.
- Use of cryptography soars.



TELEGRAM RECEIVED.

FROM 2nd from London # 5747.

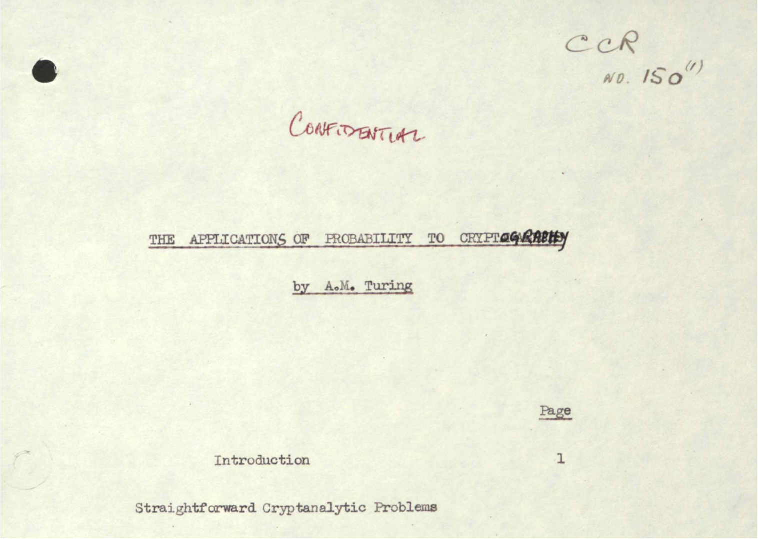
"We intend to begin on the first of February unrestricted submarine warfare. We shall endeavor in spite of this to keep the United States of America neutral. In the event of this not succeeding, we make Mexico a proposal of alliance on the following basis: make war together, make peace together, generous financial support and an understanding on our part that Mexico is to reconquer the lost territory in Texas, New Mexico, and Arizona. The settlement in detail is left to you. You will inform the President of the above most secretly as soon as the outbreak of war with the United States of America is certain and add the suggestion that he should, on his own initiative, invite Japan to immediate adherence and at the same time mediate between Japan and ourselves. Please call the President's attention to the fact that the ruthless employment of our submarines now offers the prospect of compelling England in a few months to make peace." Signed, ZIMMERMANN.

Decipherment of *Zimmermann Telegram* by British made American involvement in World War I inevitable.

Alan Turing (1912–1954)

Developed foundations of theory of computability (1936).

Still learning about Turing's contributions



CCR

NO. 150(1)

CONFIDENTIAL

THE APPLICATIONS OF PROBABILITY TO CRYPTOGRAPHY

by A.M. Turing

World War II – Enigma, Purple, JN25, Naval Enigma

- Cryptography performed by (typically, rotor) *machines*.

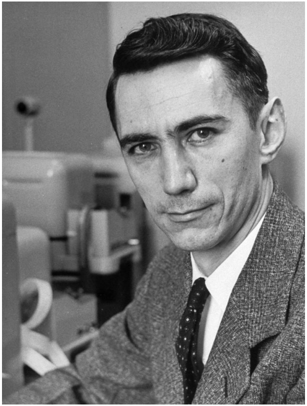
World War II – Enigma, Purple, JN25, Naval Enigma

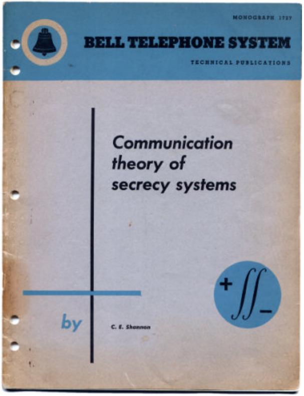
- Cryptography performed by (typically, rotor) *machines*.
- Work of Alan Turing and others at Bletchley Park, and William Friedman and others in the USA, on breaking of Axis ciphers had great success and immense impact.

World War II – Enigma, Purple, JN25, Naval Enigma

- Cryptography performed by (typically, rotor) *machines*.
- Work of Alan Turing and others at Bletchley Park, and William Friedman and others in the USA, on breaking of Axis ciphers had great success and immense impact.
- Cryptanalytic effort involved development and use of early computers (Colossus).

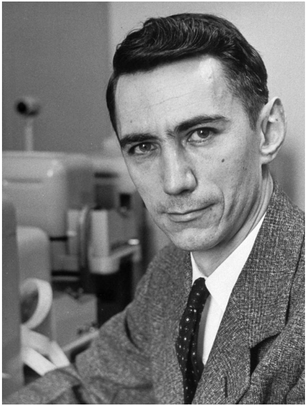
Claude Shannon (1916–2001)

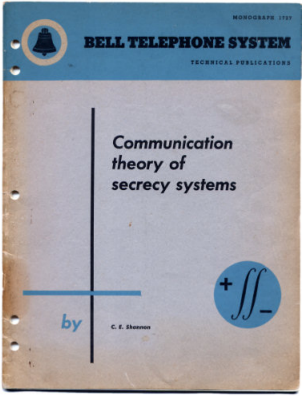




- “Communication Theory of Secrecy Systems” Sept 1945 (Bell Labs memo, classified).

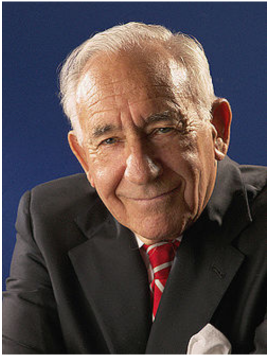
Claude Shannon (1916–2001)

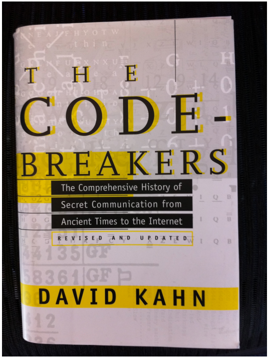




- “Communication Theory of Secrecy Systems” Sept 1945 (Bell Labs memo, classified).
- Information-theoretic in character—proves unbreakability of one-time pad. (Published 1949).

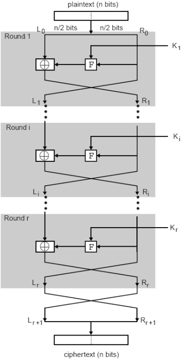
Kahn – The Codebreakers





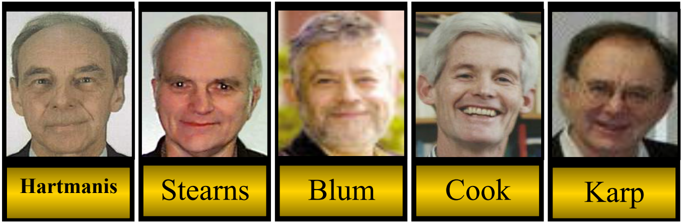
In 1967 David Kahn published
The Codebreakers—The Story of Secret Writing.
A monumental history of cryptography.
NSA attempted to suppress its publication.

DES – U.S. Data Encryption Standard (1976)



DES Designed at IBM; Horst Feistel supplied key elements of design, such as ladder structure. NSA helped, in return for keeping key size at 56 bits.(?)

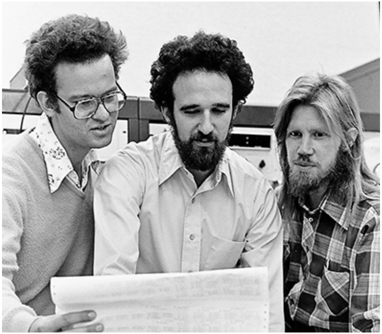
Computational Complexity



- Theory of Computational Complexity started in 1965 by Hartmanis and Stearns; expanded on by Blum, Cook, and Karp.
- Key notions: polynomial-time reductions; NP-completeness.

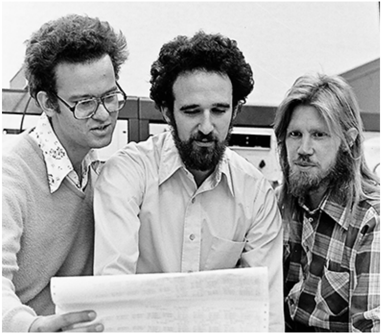
Outline

Invention of Public Key Cryptography



- Ralph Merkle, and independently Marty Hellman and Whit Diffie, invented the notion of *public-key cryptography*.

Invention of Public Key Cryptography



- Ralph Merkle, and independently Marty Hellman and Whit Diffie, invented the notion of *public-key cryptography*.
- In November 1976, Diffie and Hellman published *New Directions in Cryptography*, proclaiming

  > *"We are at the brink of a revolution in cryptography."*

Public-key encryption (as proposed by Diffie/Hellman)

- Each party A has a *public key* PK_A others can use to encrypt messages to A:

$$C = PK_A(M)$$

Public-key encryption (as proposed by Diffie/Hellman)

- Each party A has a *public key* PK_A others can use to encrypt messages to A:

$$C = PK_A(M)$$

- Each party A also has a *secret key* SK_A for decrypting a received ciphertext C:

$$M = SK_A(C)$$

Public-key encryption (as proposed by Diffie/Hellman)

- Each party A has a *public key* PK_A others can use to encrypt messages to A:

$$C = PK_A(M)$$

- Each party A also has a *secret key* SK_A for decrypting a received ciphertext C:

$$M = SK_A(C)$$

- It is easy to compute matching public/secret key pairs.

Public-key encryption (as proposed by Diffie/Hellman)

- Each party A has a *public key* PK_A others can use to encrypt messages to A:

$$C = PK_A(M)$$

- Each party A also has a *secret key* SK_A for decrypting a received ciphertext C:

$$M = SK_A(C)$$

- It is easy to compute matching public/secret key pairs.
- Publishing PK_A does not compromise SK_A! It is *computationally infeasible* to obtain SK_A from PK_A. Each public key can thus be safely listed in a public directory with the owner's name.

Digital Signatures (as proposed by Diffie/Hellman)

- Idea: sign with SK_A; verify signature with PK_A.

Digital Signatures (as proposed by Diffie/Hellman)

- Idea: sign with SK_A; verify signature with PK_A.
- A produces signature σ for message M

$$\sigma = SK_A(M)$$

Digital Signatures (as proposed by Diffie/Hellman)

- Idea: sign with SK_A; verify signature with PK_A.
- A produces signature σ for message M

$$\sigma = SK_A(M)$$

- Given PK_A, M, and σ, anyone can verify validity of signature σ by checking:

$$M \stackrel{?}{=} PK_A(\sigma)$$

Digital Signatures (as proposed by Diffie/Hellman)

- Idea: sign with SK_A; verify signature with PK_A.
- A produces signature σ for message M

$$\sigma = SK_A(M)$$

- Given PK_A, M, and σ, anyone can verify validity of signature σ by checking:

$$M \stackrel{?}{=} PK_A(\sigma)$$

- Amazing ideas!

Digital Signatures (as proposed by Diffie/Hellman)

- Idea: sign with SK_A; verify signature with PK_A.
- A produces signature σ for message M

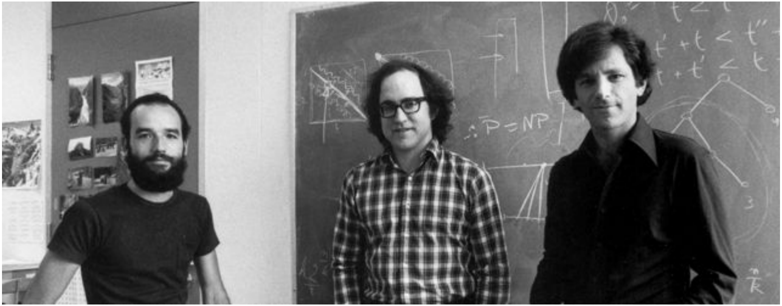
$$\sigma = SK_A(M)$$

- Given PK_A, M, and σ, anyone can verify validity of signature σ by checking:

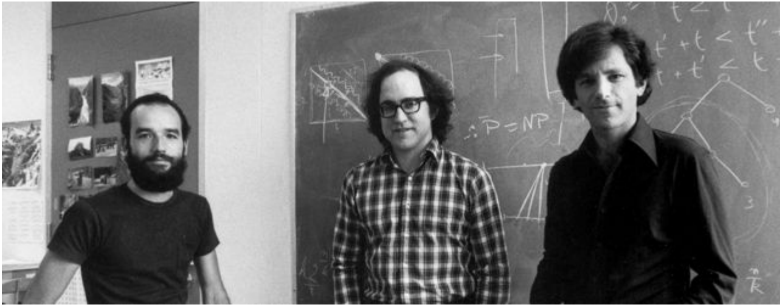
$$M \stackrel{?}{=} PK_A(\sigma)$$

- Amazing ideas!
- But they couldn't see how to implement them...

RSA (Ron Rivest, Adi Shamir, Len Adleman, 1977)

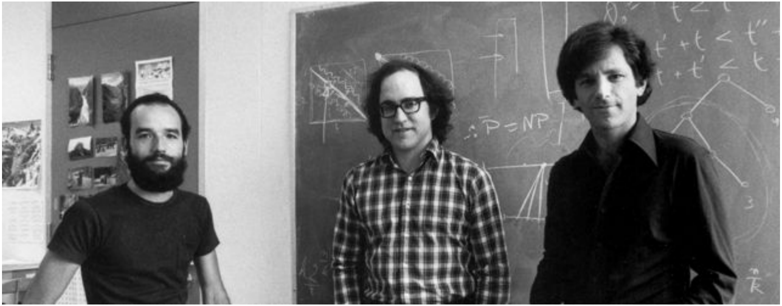


RSA (Ron Rivest, Adi Shamir, Len Adleman, 1977)



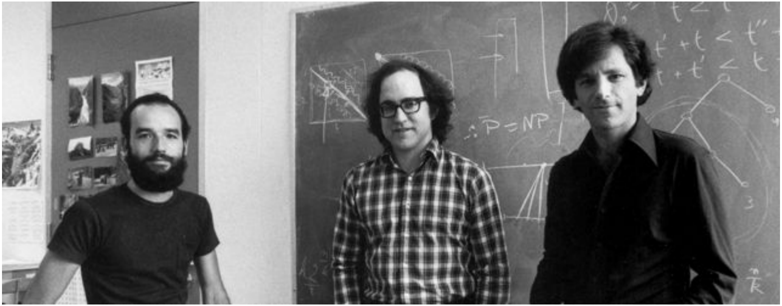
- Shamir and Adleman in Math dept.; Rivest in EECS.

RSA (Ron Rivest, Adi Shamir, Len Adleman, 1977)



- Shamir and Adleman in Math dept.; Rivest in EECS.
- Offices co-located in Laboratory for Computer Science (545 Tech. Square).

RSA (Ron Rivest, Adi Shamir, Len Adleman, 1977)



- Shamir and Adleman in Math dept.; Rivest in EECS.
- Offices co-located in Laboratory for Computer Science (545 Tech. Square).
- Adi I and proposed many methods; Len broke most of them.

Shamir's mysterious "Ski method"

Shamir's mysterious "Ski method"

- R, S, A went skiing in February 1977.

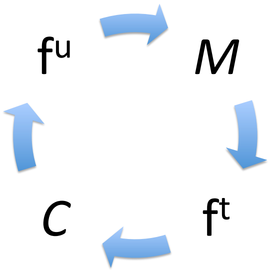
Shamir's mysterious "Ski method"

- R, S, A went skiing in February 1977.
- Shamir remembers "solving the PK problem" while skiing.

Shamir's mysterious "Ski method"

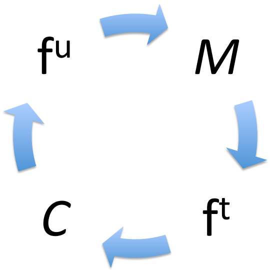
- R, S, A went skiing in February 1977.
- Shamir remembers "solving the PK problem" while skiing.
- Unfortunately, at the bottom of the run, he could no longer recall the solution...

"Almost there"–cycle with trapdoor period



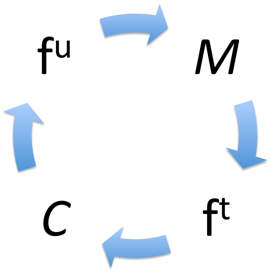
- *f* is one-way permutation with unknown (trapdoor) period *p*

"Almost there"–cycle with trapdoor period



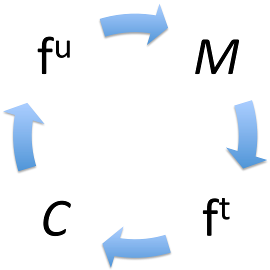
- f is one-way permutation with unknown (trapdoor) period p
- Choose t, u so that $t + u = p$

“Almost there”–cycle with trapdoor period



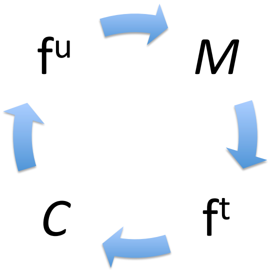
- f is one-way permutation with unknown (trapdoor) period p
- Choose t, u so that $t + u = p$
- f^t, f^u easily computed

“Almost there”–cycle with trapdoor period



- f is one-way permutation with unknown (trapdoor) period p
- Choose t, u so that $t + u = p$
- f^t, f^u easily computed
- Encrypt: $c = f^t(m)$

"Almost there"–cycle with trapdoor period



- f is one-way permutation with unknown (trapdoor) period p
- Choose t, u so that $t + u = p$
- f^t, f^u easily computed
- Encrypt: $c = f^t(m)$
- Decrypt: $m = f^u(c)$

Seder

- Seder dinner April 1977 at home of Anni Bruss.

Seder

- Seder dinner April 1977 at home of Anni Bruss.
- "*In vino veritas*" (Pliny $\approx$ AD 50)

Seder

- Seder dinner April 1977 at home of Anni Bruss.
- "*In vino veritas*" (Pliny $\approx$ AD 50)

- Manichewitz wine + permutation polynomials + factoring...

RSA method

- Security relies (in part) on inability to factor product n of two large primes p, q.

RSA method

- Security relies (in part) on inability to factor product n of two large primes p, q.
- $PK = (n, e)$ where $n = pq$ and $\gcd(e, \phi(n)) = 1$

RSA method 

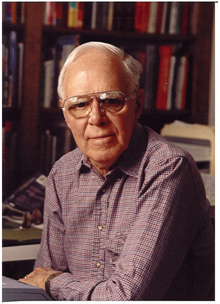
- Security relies (in part) on inability to factor product n of two large primes p, q.
- $PK = (n, e)$ where $n = pq$ and $\gcd(e, \phi(n)) = 1$
- $SK = d$ where $de = 1 \mod \phi(n)$

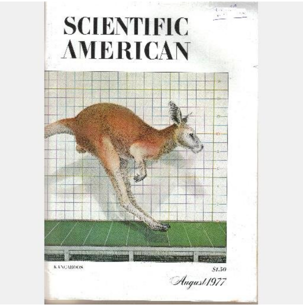
RSA method

- Security relies (in part) on inability to factor product n of two large primes p, q.
- $PK = (n, e)$ where $n = pq$ and $\gcd(e, \phi(n)) = 1$
- $SK = d$ where $de = 1 \mod \phi(n)$
- Encryption/decryption (or signing/verify) are simple:

$$\begin{aligned} C &= PK(M) = M^e \mod n \\ M &= SK(C) = C^d \mod n \end{aligned}$$

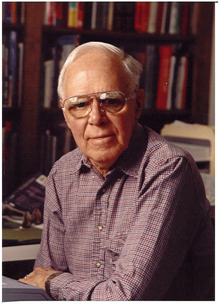
Martin Gardner column and RSA-129 challenge

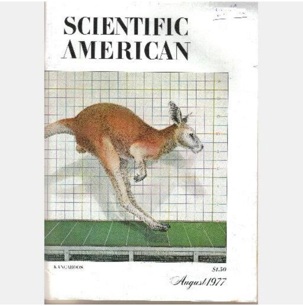




- Described public-key and RSA cryptosystem in his Scientific American column, *Mathematical Games*

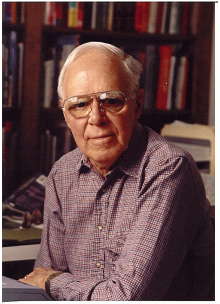
Martin Gardner column and RSA-129 challenge

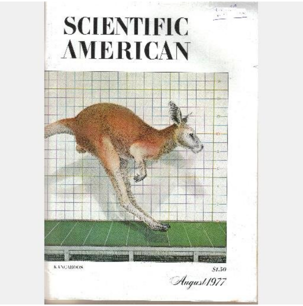




- Described public-key and RSA cryptosystem in his Scientific American column, *Mathematical Games*
- Offered copy of RSA technical memo.

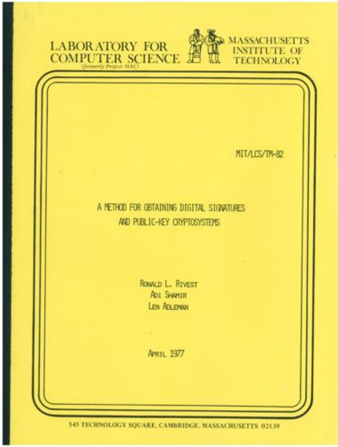
Martin Gardner column and RSA-129 challenge

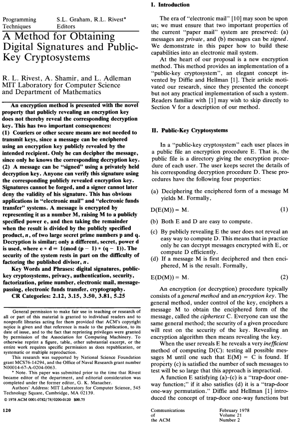




- Described public-key and RSA cryptosystem in his Scientific American column, *Mathematical Games*
- Offered copy of RSA technical memo.
- Offered $100 to first person to break challenge ciphertext based on 129-digit product of primes. (Our) estimated time to solution: 40 quadrillion years

Publication of RSA memo and paper





Programming Techniques — S.L. Graham, R.L. Rivest* Editors

A Method for Obtaining Digital Signatures and Public-Key Cryptosystems

R. L. Rivest, A. Shamir, and L. Adleman
MIT Laboratory for Computer Science
and Department of Mathematics

An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences:
(1) Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key.
(2) A message can be "signed" using a privately held decryption key. Anyone can verify this signature using the corresponding publicly revealed encryption key. Signatures cannot be forged, and a signer cannot later deny the validity of his signature. This has obvious applications in "electronic mail" and "electronic funds transfer" systems. A message is encrypted by representing it as a number M, raising M to a publicly specified power e, and then taking the remainder when the result is divided by the publicly specified product, *n*, of two large secret prime numbers p and q. Decryption is similar; only a different, secret, power d is used, where $e * d \equiv 1 (\mathrm{mod}\ (p-1) * (q-1))$. The security of the system rests in part on the difficulty of factoring the published divisor, *n*.

Key Words and Phrases: digital signatures, public-key cryptosystems, privacy, authentication, security, factorization, prime number, electronic mail, message-passing, electronic funds transfer, cryptography.
CR Categories: 2.12, 3.15, 3.50, 3.81, 5.25

This research was supported by National Science Foundation grant MCS76-14294, and the Office of Naval Research grant number N00014-67-A-0204-0063.

* Note. This paper was submitted prior to the time that Rivest became editor of the department, and editorial consideration was completed under the former editor, G. K. Manacher.

Authors' Address: MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, MA 02139.

I. Introduction

The era of "electronic mail" [10] may soon be upon us; we must ensure that two important properties of the current "paper mail" system are preserved: (a) messages are *private*, and (b) messages can be *signed*. We demonstrate in this paper how to build these capabilities into an electronic mail system.

At the heart of our proposal is a new encryption method. This method provides an implementation of a "public-key cryptosystem", an elegant concept invented by Diffie and Hellman [1]. Their article motivated our research, since they presented the concept but not any practical implementation of such a system. Readers familiar with [1] may wish to skip directly to Section V for a description of our method.

II. Public-Key Cryptosystems

In a "public-key cryptosystem" each user places in a public file an encryption procedure E. That is, the public file is a directory giving the encryption procedure of each user. The user keeps secret the details of his corresponding decryption procedure D. These procedures have the following four properties:

(a) Deciphering the enciphered form of a message M yields M. Formally,

$$D(E(M)) = M. \qquad (1)$$

(b) Both E and D are easy to compute.

(c) By publicly revealing E the user does not reveal an easy way to compute D. This means that in practice only he can decrypt messages encrypted with E, or compute D efficiently.

(d) If a message M is first deciphered and then enciphered, M is the result. Formally,

$$E(D(M)) = M. \qquad (2)$$

An encryption (or decryption) procedure typically consists of a *general method* and an *encryption key*. The general method, under control of the key, enciphers a message M to obtain the enciphered form of the message, called the *ciphertext* C. Everyone can use the same general method; the security of a given procedure will rest on the security of the key. Revealing an encryption algorithm then means revealing the key.

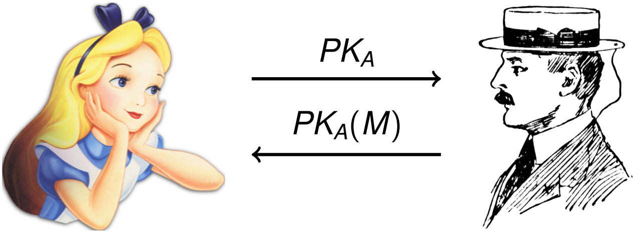
When the user reveals E he reveals a very *inefficient* method of computing D(C): testing all possible messages M until one such that E(M) = C is found. If property (c) is satisfied the number of such messages to test will be so large that this approach is impractical.

A function E satisfying (a)–(c) is a "trap-door one-way function;" if it also satisfies (d) it is a "trap-door one-way permutation." Diffie and Hellman [1] introduced the concept of trap-door one-way functions but

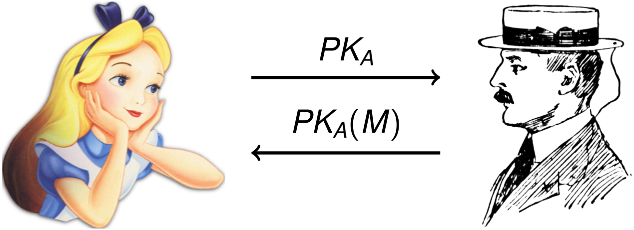
LCS-82 Technical Memo (April 1977)
CACM article (Feb 1978)

Alice and Bob (1977, in RSA paper)

Alice and Bob (1977, in RSA paper)



Alice and Bob (1977, in RSA paper)



Alice and Bob now have a life of their own—they appear in hundreds of crypto papers, in `xkcd`, and even have their own Wikipedia page:

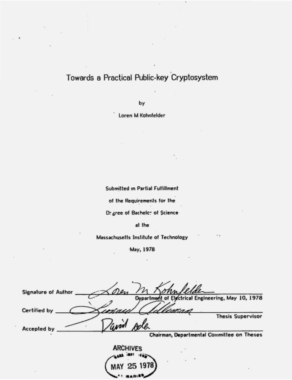
Independent Invention of Public-Key Revealed



In 1999 GCHQ announced that James Ellis, Clifford Cocks, and Malcolm Williamson had invented public-key cryptography, the “RSA” algorithm, and “Diffie-Hellman key exchange” in the 1970’s, before their invention outside.

Outline

Loren Kohnfelder – Invention of Digital Certificates


Towards a Practical Public-key Cryptosystem

by

Loren M Kohnfelder

Submitted in Partial Fulfillment
of the Requirements for the
Degree of Bachelor of Science
at the
Massachusetts Institute of Technology
May, 1978

Signature of Author ______________________
Department of Electrical Engineering, May 10, 1978

Certified by ______________________
Thesis Supervisor

Accepted by ______________________
Chairman, Departmental Committee on Theses

ARCHIVES
MAY 25 1978

- Loren Kohnfelder's B.S. thesis (MIT 1978, supervised by Len Adleman), proposed notion of *digital certificate*—a digitally signed message attesting to another party's public key.

RSA on a chip (1980)

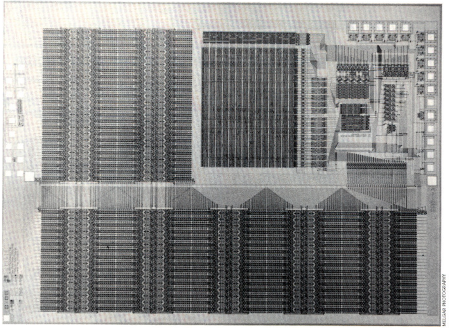


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

- MIT started VLSI effort.

RSA on a chip (1980)

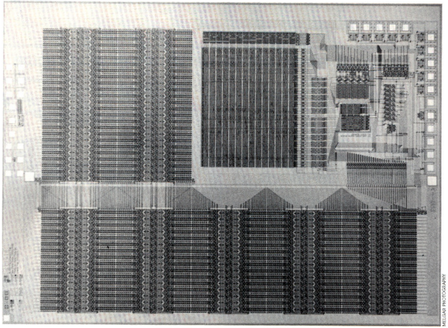


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

LAMBDA *Fourth Quarter 1980* 17

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:

RSA on a chip (1980)

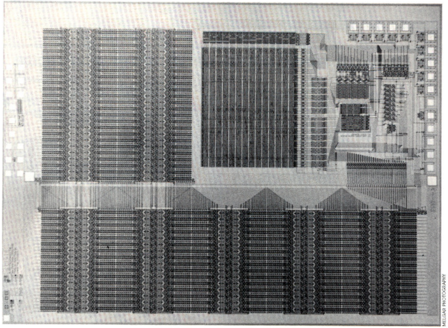


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:
  - 512-bit bignum processor

RSA on a chip (1980)

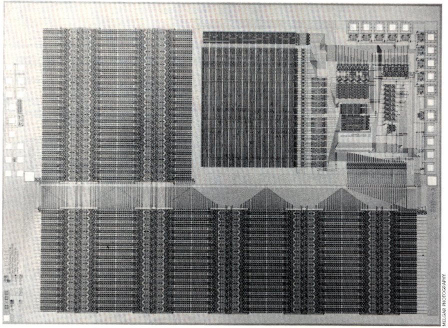


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

LAMBDA *Fourth Quarter 1980* 17

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:
  - 512-bit bignum processor
  - RSA key generation (including prime-finding)

RSA on a chip (1980)

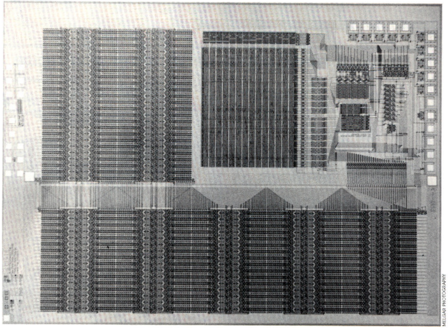


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

LAMBDA *Fourth Quarter 1980* 17

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:
  - 512-bit bignum processor
  - RSA key generation (including prime-finding)
  - Pollard’s “rho” factorization method

RSA on a chip (1980)

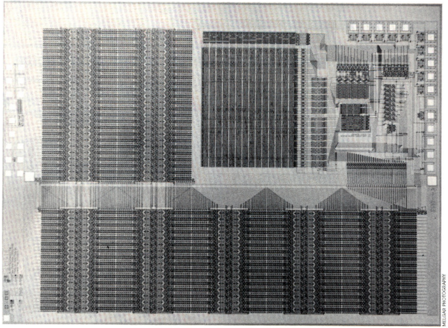


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

LAMBDA *Fourth Quarter 1980* 17

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:
  - 512-bit bignum processor
  - RSA key generation (including prime-finding)
  - Pollard’s “rho” factorization method
  - 40,000 transistors; 5.5mm x 8mm chip.

RSA on a chip (1980)

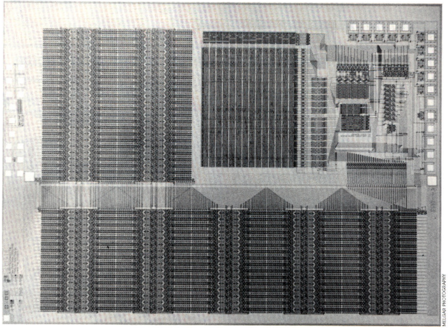


FIGURE 3. The RSA chip contains 40,000 transistors and measures 5.5 mm by 8 mm.

LAMBDA *Fourth Quarter 1980* 17

- MIT started VLSI effort.
- R, S, A designed "RSA chip" and fabbed prototype:
  - 512-bit bignum processor
  - RSA key generation (including prime-finding)
  - Pollard's "rho" factorization method
  - 40,000 transistors; 5.5mm x 8mm chip.
- Fabrication was buggy/unreliable.

IACR—International Assn. for Cryptologic Research

- Established 1982 by David Chaum, myself, and others, to promote academic research in cryptology.
- Sponsors three major conferences/year (Crypto, Eurocrypt, Asiacrypt) and four workshops; about 200 papers/year, plus another 600/year posted on web. Publishes J. Cryptography
- Around 1600 members, (25% students), from 74 countries, 54 Fellows.

Theoretical Foundations of Security

- “Probabilistic Encryption” Shafi Goldwasser, Silvio Micali (1982) (Encryption should be *randomized*!)

Theoretical Foundations of Security

- "Probabilistic Encryption" Shafi Goldwasser, Silvio Micali (1982) (Encryption should be *randomized*!)
- "A Digital Signature Scheme Secure Against Adaptive Chosen Message Attacks" Goldwasser, Micali, Rivest (1988) (Uses well-defined *game* to define security objective.)

RC4 stream cipher (Rivest, 1987)

- RC4 is the most widely used software stream cipher

RC4 stream cipher (Rivest, 1987)

- RC4 is the most widely used software stream cipher
- Not public-key; xors stream of pseudo-random bytes with plaintext to derive ciphertext.

RC4 stream cipher (Rivest, 1987)

- RC4 is the most widely used software stream cipher
- Not public-key; xors stream of pseudo-random bytes with plaintext to derive ciphertext.
- Extremely simple and fast: uses array $S[0..255]$ to keep a permutation of 0..255, initialized using secret key, and uses two pointers i,j into S.

  To output a pseudo-random byte:

```
i = (i + 1) mod 256
j = (j + S[i]) mod 256
swap S[i] and S[j]
Output S[(S[i] + S[j]) mod 256]
```

RC4 stream cipher (Rivest, 1987)

- RC4 is the most widely used software stream cipher
- Not public-key; xors stream of pseudo-random bytes with plaintext to derive ciphertext.
- Extremely simple and fast: uses array $S[0..255]$ to keep a permutation of 0..255, initialized using secret key, and uses two pointers i,j into S.

  To output a pseudo-random byte:

  ```
  i = (i + 1) mod 256
  j = (j + S[i]) mod 256
  swap S[i] and S[j]
  Output S[(S[i] + S[j]) mod 256]
  ```
- Used in: WEP, BitTorrent, SSL, Kerberos, PDF, Skype, ...

RC4 stream cipher (Rivest, 1987)

- RC4 is the most widely used software stream cipher
- Not public-key; xors stream of pseudo-random bytes with plaintext to derive ciphertext.
- Extremely simple and fast: uses array $S[0..255]$ to keep a permutation of 0..255, initialized using secret key, and uses two pointers i,j into S.

  To output a pseudo-random byte:

```
i = (i + 1) mod 256
j = (j + S[i]) mod 256
swap S[i] and S[j]
Output S[(S[i] + S[j]) mod 256]
```

- Used in: WEP, BitTorrent, SSL, Kerberos, PDF, Skype, ...
- Showing its age (statistical attacks)...

Spritz – RC4 replacement (w/ J. Schuldt, 2014)

RC4()

$i = i + 1$
$j = j + S[i]$

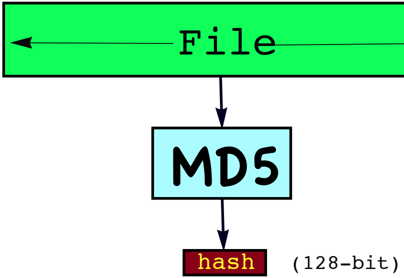
$\text{SWAP}(S[i], S[j])$
$z = S[S[i] + S[j]]$
return z

SPRITZ()

$i = i + 1$
$j = k + S[j + S[i]]$
$k = i + k + S[j]$
$\text{SWAP}(S[i], S[j])$
$z = S[j + S[i + S[z + k]]]$
return z

- Spritz code found by computer search.
- About 50% longer and 4X slower (unoptimized).
- Uses new register k as well RC4 registers i, j; output register z also used in feedback.
- 2^{81} samples seem necessary to distinguish SPRITZ-256 from random. (Compare: 2^{41} for RC4.)

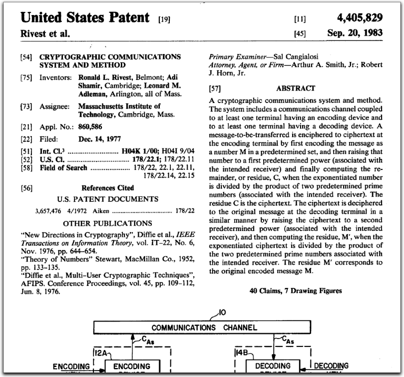
MD5 Cryptographic Hash Function (Rivest, 1991)



- MD5 proposed as pseudo-random function mapping files to 128-bit fingerprints. (variant of earlier MD4; ARX-style)
- Collision-resistance was a design goal – it should be infeasible to find two files with the same fingerprint.
- Many, many uses (e.g. in digital signatures) – very widely used, and a model for many other later hash function designs.

Outline

U.S. Patent 4,405,829



United States Patent [19] [11] **4,405,829**

Rivest et al. [45] **Sep. 20, 1983**

[54] **CRYPTOGRAPHIC COMMUNICATIONS SYSTEM AND METHOD**

[75] Inventors: **Ronald L. Rivest,** Belmont; **Adi Shamir,** Cambridge; **Leonard M. Adleman,** Arlington, all of Mass.

[73] Assignee: **Massachusetts Institute of Technology,** Cambridge, Mass.

[21] Appl. No.: **860,586**

[22] Filed: **Dec. 14, 1977**

[51] **Int. Cl.³** **H04K 1/00;** H04I 9/04

[52] **U.S. Cl.** **178/22.1;** 178/22.11

[58] **Field of Search** 178/22, 22.1, 22.11, 178/22.14, 22.15

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,657,476 4/1972 Aiken 178/22

OTHER PUBLICATIONS

"New Directions in Cryptography", Diffie et al., *IEEE Transactions on Information Theory*, vol. IT–22, No. 6, Nov. 1976, pp. 644–654.

"Theory of Numbers" Stewart, MacMillan Co., 1952, pp. 133–135.

"Diffie et al., Multi-User Cryptographic Techniques", AFIPS. Conference Proceedings, vol. 45, pp. 109–112, Jun. 8, 1976.

Primary Examiner—Sal Cangialosi
Attorney, Agent, or Firm—Arthur A. Smith, Jr.; Robert J. Horn, Jr.

[57] **ABSTRACT**

A cryptographic communications system and method. The system includes a communications channel coupled to at least one terminal having an encoding device and to at least one terminal having a decoding device. A message-to-be-transferred is enciphered to ciphertext at the encoding terminal by first encoding the message as a number M in a predetermined set, and then raising that number to a first predetermined power (associated with the intended receiver) and finally computing the remainder, or residue, C, when the exponentiated number is divided by the product of two predetermined prime numbers (associated with the intended receiver). The residue C is the ciphertext. The ciphertext is deciphered to the original message at the decoding terminal in a similar manner by raising the ciphertext to a second predetermined power (associated with the intended receiver), and then computing the residue, M', when the exponentiated ciphertext is divided by the product of the two predetermined prime numbers associated with the intended receiver. The residue M' corresponds to the original encoded message M.

40 Claims, 7 Drawing Figures

10
COMMUNICATIONS CHANNEL
C_{As}
12A
ENCODING
ENCODING
C_{As}
14B
DECODING
DECODING

Filed December 1977 (MIT TLO)
Issued September 1983

RSA the company (1983)

RSA the company (1983)

- Jim Bidzos joined in 1986

RSA the company (1983)

- Jim Bidzos joined in 1986
- Lotus (1987), Motorola, Apple, Novell, Netscape, Microsoft, ...

RSA the company (1983)

- Jim Bidzos joined in 1986
- Lotus (1987), Motorola, Apple, Novell, Netscape, Microsoft, ...
- RSA Conference series (1991)

RSA the company (1983)

- Jim Bidzos joined in 1986
- Lotus (1987), Motorola, Apple, Novell, Netscape, Microsoft, ...
- RSA Conference series (1991)
- Verisign spun out in 1995

  1.3 billion certificate status checks/day

  65 billion DNS requests/day (DNSSEC coming)

RSA the company (1983)

- Jim Bidzos joined in 1986
- Lotus (1987), Motorola, Apple, Novell, Netscape, Microsoft, ...
- RSA Conference series (1991)
- Verisign spun out in 1995
  - 1.3 billion certificate status checks/day
  - 65 billion DNS requests/day (DNSSEC coming)
- RSA acquired by Security Dyamics in 1996, now part of EMC.

World Wide Web (Sir Tim Berners-Lee, 1990)

- Just as radio did, this new communication medium, the World-Wide Web, drove demand for cryptography to new heights.
- Cemented transition of cryptography from primarily military to primarily commercial.

Outline

U.S. cryptography policy evolves

- U.S. government initially tried to control and limit public-sector research and use of cryptography
- Attempt to chill research via ITAR (1977)
- MIT "Changing Nature of Information" Committee (1981; Dertouzos, Low, Rosenblith, Deutch,Rivest,...)

MIT Committee Seeks Cryptography Policy

Questions of who should do research on cryptography and how results should be disseminated are the first order of business

Within the next 10 years, networks consisting of tens of thousands of computers will connect businesses, corpora- ... quences for individuals and for society if computers continue to be connected, as they are now, according to local deci- ... easy to send computer programs between connected machines and to instruct a program to search for, select, ...

Science, 13 Mar 1981

U.S. cryptography policy evolves

- U.S. government tried to mandate availability of all encryption keys via “key escrow” and/or “Clipper Chip” (1993)

U.S. cryptography policy evolves

- U.S. government tried to mandate availability of all encryption keys via “key escrow” and/or “Clipper Chip” (1993)

U.S. cryptography policy evolves

- U.S. government tried to mandate availability of all encryption keys via “key escrow” and/or “Clipper Chip” (1993)

- With defeat of “Clipper Chip”, it seemed “crypto wars” were over; strong crypto was recognized as necessary for commerce and for national security...

U.S. cryptography policy evolves

- U.S. government tried to mandate availability of all encryption keys via “key escrow” and/or “Clipper Chip” (1993)
- With defeat of “Clipper Chip”, it seemed “crypto wars” were over; strong crypto was recognized as necessary for commerce and for national security...
- Recently, this issue has re-surfaced...

Outline

Factorization of RSA-129 (April 1994)

- RSA-129 =

```
1143816257578888676692357799761466120102182 9
6721242362562561842935706935245733897830597 1
23563958705058989075147599290026879543541
```

Factorization of RSA-129 (April 1994)

- RSA-129 =

  ```
  114381625757888867669235779976146612010218296
  7212423625625618429357069352457338978305971
  23563958705058989075147599290026879543541
  ```

- Derek Atkins, Michael Graff, Arjen Lenstra, Paul Leyland: RSA-129 =

  ```
  3490529510847650949147849619903898133417764
  6384933878439908205 77 x
  3276913299326670954996198819083446141317764
  2967992942539798288533
  ```

Factorization of RSA-129 (April 1994)

- RSA-129 =

  ```
  114381625757888867669235779976146612010218296721242362562561842935706935245733897830597123563958705058989075147599290026879543541
  ```

- Derek Atkins, Michael Graff, Arjen Lenstra, Paul Leyland: RSA-129 =

  ```
  3490529510847650949147849619903898133417764638493387843990820577 x
  32769132993266709549961988190834461413177642967992942539798288533
  ```

- 8 months work by about 600 volunteers from more than 20 countries; 5000 MIPS-years.

Factorization of RSA-129 (April 1994)

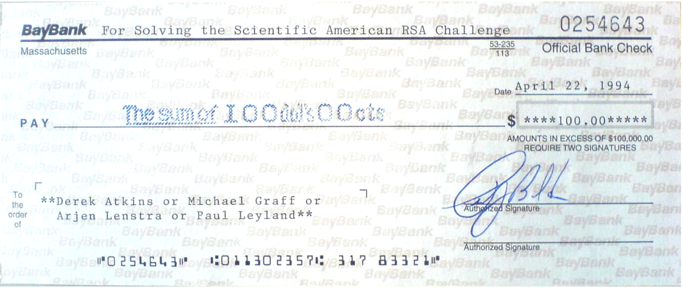
- RSA-129 =

  ```
  114381625757888867669235779976146612010218296
  721242362562561842935706935245733897830597123
  56395870505898907514759929002687954354 1
  ```

- Derek Atkins, Michael Graff, Arjen Lenstra,
  Paul Leyland: RSA-129 =

  ```
  3490529510847650949147849619903898133417764
  6384933878439908205 77 x
  3276913299326670954996198819083446141317764
  296799294253979828853 3
  ```

- 8 months work by about 600 volunteers from more than 20 countries; 5000 MIPS-years.
- secret message:
  The Magic Words Are Squeamish Ossifrage


BayBank For Solving the Scientific American RSA Challenge 0254643

Massachusetts 53-235/113 Official Bank Check

Date April 22, 1994

PAY The sum of 100 dols 00 cts $ ****100.00*****

AMOUNTS IN EXCESS OF $100,000.00 REQUIRE TWO SIGNATURES

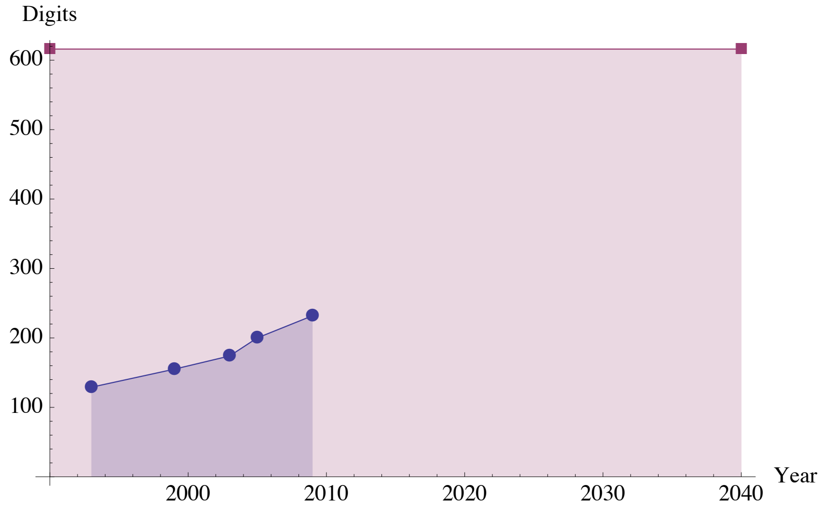
To the order of **Derek Atkins or Michael Graff or Arjen Lenstra or Paul Leyland**

Authorized Signature

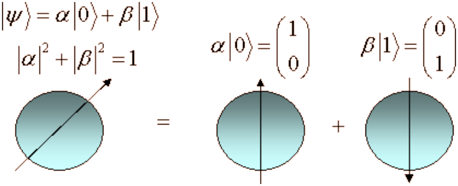
Authorized Signature

0254643 011302357 317 83321

Factoring Records

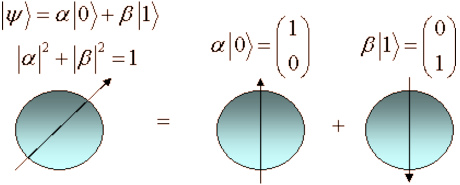


Factoring on a Quantum Computer?



In 1994, Peter Shor invented a fast factorization algorithm that runs on a (hypothetical) *quantum computer* and works by determining multiplicative period of elements mod *n*.

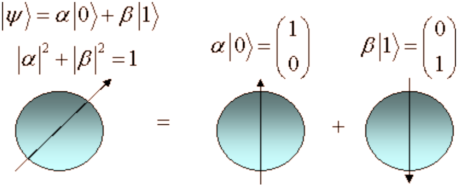
Factoring on a Quantum Computer?



In 1994, Peter Shor invented a fast factorization algorithm that runs on a (hypothetical) *quantum computer* and works by determining multiplicative period of elements mod n.

- In 2001, researchers at IBM used this algorithm on a (real) quantum computer to factor $15 = 3 \times 5$.

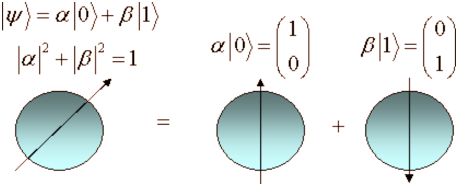
Factoring on a Quantum Computer?



In 1994, Peter Shor invented a fast factorization algorithm that runs on a (hypothetical) *quantum computer* and works by determining multiplicative period of elements mod n.

- In 2001, researchers at IBM used this algorithm on a (real) quantum computer to factor $15 = 3 \times 5$.
- Recently (Dattani, 2014): 291311 = 557 x 523

Factoring on a Quantum Computer?



In 1994, Peter Shor invented a fast factorization algorithm that runs on a (hypothetical) *quantum computer* and works by determining multiplicative period of elements mod *n*.

- In 2001, researchers at IBM used this algorithm on a (real) quantum computer to factor $15 = 3 \times 5$.
- Recently (Dattani, 2014): 291311 = 557 x 523
- Dark clouds on horizon for RSA?

Hash Function Attacks

- In 2004 Xiaoyun Wang and colleagues found a way to produce *collisions* for MD5:

$$\mathtt{MD5}(\mathit{file}1) = \mathtt{MD5}(\mathit{file}2) \quad !!!$$

Also for SHA-1 and many other hash functions.
Major break!!

Hash Function Attacks

- In 2004 Xiaoyun Wang and colleagues found a way to produce *collisions* for MD5:

  $$\texttt{MD5}(\textit{file}1) = \texttt{MD5}(\textit{file}2) \quad \text{!!!}$$

  Also for SHA-1 and many other hash functions. Major break!!
- So NIST ran a competition for new hash function standard (SHA-3 = Keccak).

Outline

Many new research problems and directions

- secret-sharing
- anonymity
- commitments
- multi-party protocols
- elliptic curves
- crypto hardware
- key leakage
- proxy encryption
- crypto for smart cards
- password-based keys
- random oracles
- oblivious transfer
- ...

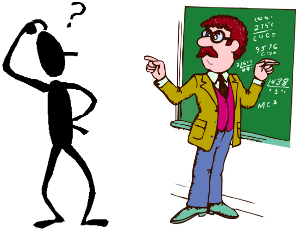
- zero-knowledge proofs
- payment systems
- voting systems
- homomorphic encryption
- lattice-based crypto
- private information retrieval
- public-key infrastructure
- concurrent protocols
- randomness extractors
- tweakable encryption
- differential cryptanalysis
- identity-based encryption
- ...

Many new research problems and directions

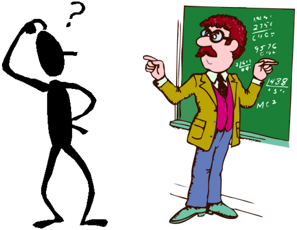
- secret-sharing
- anonymity
- commitments
- multi-party protocols
- elliptic curves
- crypto hardware
- key leakage
- proxy encryption
- crypto for smart cards
- password-based keys
- random oracles
- oblivious transfer
- ...

- zero-knowledge proofs
- payment systems
- voting systems
- homomorphic encryption
- lattice-based crypto
- private information retrieval
- public-key infrastructure
- concurrent protocols
- randomness extractors
- tweakable encryption
- differential cryptanalysis
- identity-based encryption
- ...

Zero-Knowledge Proofs

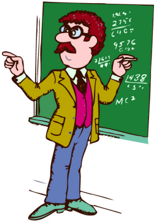


Zero-Knowledge Proofs



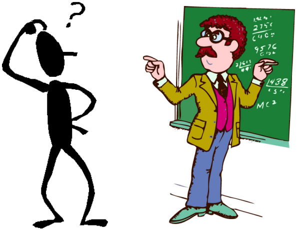
I can convince you

Zero-Knowledge Proofs



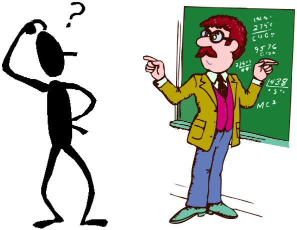
I can convince you
I know a solution
to a hard problem

Zero-Knowledge Proofs



I can convince you
I know a solution
to a hard problem
while telling you nothing
about my solution

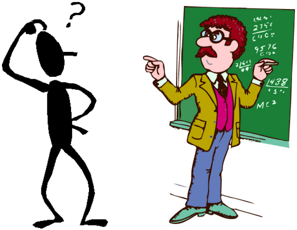
Zero-Knowledge Proofs



I can convince you
I know a solution
to a hard problem
while telling you nothing
about my solution
even if you are very skeptical!

Goldwasser, Micali, Rackoff (1985)
Goldreich, Micali, Wigderson (1986)

Zero-Knowledge Proofs



I can convince you
I know a solution
to a hard problem
while telling you nothing
about my solution
even if you are very skeptical!

Goldwasser, Micali, Rackoff (1985)
Goldreich, Micali, Wigderson (1986)

An enormously useful capability!

Payment Systems

- *Probabilistic payments* (Micali and Rivest, 2001). "Peppercoin" payments. Paying you ten cents is like paying you one dollar with probability 1/10.

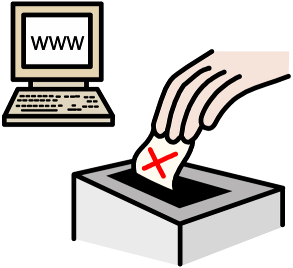
Payment Systems

- *Probabilistic payments* (Micali and Rivest, 2001). "Peppercoin" payments. Paying you ten cents is like paying you one dollar with probability 1/10.
- *Bitcoin* (Nakamoto, 2009). The "blockchain" for decentralized consensus.

Payment Systems

- *Probabilistic payments* (Micali and Rivest, 2001). “Peppercoin” payments. Paying you ten cents is like paying you one dollar with probability 1/10.
- *Bitcoin* (Nakamoto, 2009). The “blockchain” for decentralized consensus.
- Ethereum, Dogecoin, Litecoin, Zero-cash, ...

Voting Systems

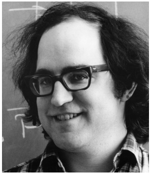
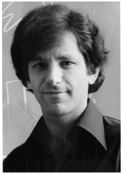


New "end-to-end" cryptographic voting systems (Chaum, Neff, Benaloh, Ryan, Rivest, Adida, ...):

- all ballots posted on web (encrypted)
- voters verify their votes are correct (while preventing vote-selling and coercion)
- anyone can verify final tally
- may be done with paper ballots

Cryptography *increases* transparency and verifiability!

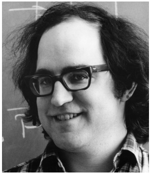
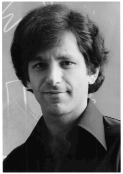
Fully Homomorphic Encryption

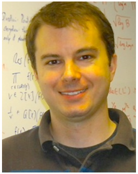
   ?

- In 1978, Rivest, Adleman, and Dertouzos asked,

  "Can one compute on encrypted data,
  while keeping it encrypted?"

Fully Homomorphic Encryption

?

!

- In 1978, Rivest, Adleman, and Dertouzos asked,

  "Can one compute on encrypted data, while keeping it encrypted?"

- In 2009, Craig Gentry (Stanford,IBM) gave solution based on use of lattices. If efficiency can be greatly improved, could be huge implications (e.g. for cloud computing).

Outline

Crypto Wars 2.0

- Apple / FBI iphone debate...
- Should LE have ability to unlock any iPhone or encryption content?
- Read “Keys Under Doormats” report (Abelson et al. 2015)

Outline

Challenges

- Make more crypto theory results practical

Challenges

- Make more crypto theory results practical
- Is factoring really hard?

Challenges

- Make more crypto theory results practical
- Is factoring really hard?
- Minimize assumptions; evaluate assumptions

Challenges

- Make more crypto theory results practical
- Is factoring really hard?
- Minimize assumptions; evaluate assumptions
- Show $P \neq NP$!

Challenges

- Make more crypto theory results practical
- Is factoring really hard?
- Minimize assumptions; evaluate assumptions
- Show $P \neq NP$!
- Is quantum computing practical?

Challenges

- Make more crypto theory results practical
- Is factoring really hard?
- Minimize assumptions; evaluate assumptions
- Show $P \neq NP$!
- Is quantum computing practical?
- Give Alice and Bob smartphones!

Challenges

- Make more crypto theory results practical
- Is factoring really hard?
- Minimize assumptions; evaluate assumptions
- Show $P \neq NP$!
- Is quantum computing practical?
- Give Alice and Bob smartphones!
- Ground crypto practice better in vulnerable computer systems; prepare better for worst-case scenarios.

Conclusions

- Cryptography is not the solution to all of our cybersecurity problems, but it is an essential component of any solution.

Conclusions

- Cryptography is not the solution to all of our cybersecurity problems, but it is an essential component of any solution.
- Research in cryptography is a fascinating blend of mathematics, statistics, theoretical computer science, electrical engineering, and psychology.

Conclusions

- Cryptography is not the solution to all of our cybersecurity problems, but it is an essential component of any solution.
- Research in cryptography is a fascinating blend of mathematics, statistics, theoretical computer science, electrical engineering, and psychology.
- While we have accomplished a lot in a few decades, much remains to be done.

Conclusions

- Cryptography is not the solution to all of our cybersecurity problems, but it is an essential component of any solution.
- Research in cryptography is a fascinating blend of mathematics, statistics, theoretical computer science, electrical engineering, and psychology.
- While we have accomplished a lot in a few decades, much remains to be done.
- Like Alice and Bob, cryptography is here to stay.

Conclusions

- Cryptography is not the solution to all of our cybersecurity problems, but it is an essential component of any solution.
- Research in cryptography is a fascinating blend of mathematics, statistics, theoretical computer science, electrical engineering, and psychology.
- While we have accomplished a lot in a few decades, much remains to be done.
- Like Alice and Bob, cryptography is here to stay.
- Cryptography is fun!

Thank You!

The 2016 Cybersecurity Speaker Series

Thank you for joining.

For more information, visit brown.edu/cybersecurity

STRATEGY IS THE BEST SECURITY.

The Executive Master in Cybersecurity BROWN