On the growth of cryptography

Ronald L. Rivest

Vannevar Bush Professor of EECS
MIT, Cambridge, MA

Simons Institute Cryptography Program
Historical Papers Seminar Series
U.C. Berkeley
June 3, 2015

\(^1\)many slides from my 2011 MIT Killian award lecture
Outline

Some pre-1976 context
Invention of Public-Key Crypto and RSA
Early steps
The cryptography business
Crypto policy
Attacks
More New Directions
What Next?
Conclusions
Outline

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
There are infinitely many primes:
2, 3, 5, 7, 11, 13, ...
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”):
\[ \text{gcd}(12, 30) = 6 \]
An unknown *period* (the circumference of the scytale) is the secret key, shared by sender and receiver.
Fermat’s Little Theorem (1640):
For any prime $p$ and any $a$, $1 \leq a < p$:

$$a^{p-1} = 1 \pmod{p}$$
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 $\phi(n) = \# \text{ of } x < n \text{ such that } \gcd(x, n) = 1$. 
Carl Friedrich Gauss (1777-1855)

Published *Disquisitiones Arithmeticae* at age 21
Published *Disquisitiones Arithmeticae* 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)
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.”
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.

- Use of cryptography soars. Decipherment of the Zimmermann Telegram by British made American involvement in World War I inevitable.
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.

Decipherment of Zimmermann Telegram by British made American involvement in World War I inevitable.
Developed foundations of theory of computability (1936).
THE APPLICATIONS OF PROBABILITY TO CRYPTOGRAPHY

by A.M. Turing

Introduction

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

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

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
Invention of Public Key Cryptography

- Ralph Merkle, and independently Marty Hellman and Whit Diffie, invented the notion 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 $\sigma$ 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 $\sigma$ for message $M$
    $$\sigma = SK_A(M)$$
  - Given $PK_A$, $M$, and $\sigma$, anyone can verify validity of signature $\sigma$ by checking:
    $$M \overset{?}{=} PK_A(\sigma)$$
Digital Signatures (as proposed by Diffie/Hellman)

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

\[ \sigma = SK_A(M) \]

- Given $PK_A$, $M$, and $\sigma$, anyone can verify validity of signature $\sigma$ by checking:

\[ M \overset{?}{=} PK_A(\sigma) \]

- Amazing ideas!
Digital Signatures (as proposed by Diffie/Hellman)

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

$$\sigma = SK_A(M)$$

- Given $PK_A$, $M$, and $\sigma$, anyone can verify validity of signature $\sigma$ by checking:

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

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

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...
Shamir’s mysterious “Ski method”


Shamir remembers “solving the PK problem” while skiing. Unfortunately, at the bottom of the run, he could no longer recall the solution...
Shamir’s mysterious “Ski method”

- Shamir remembers “solving the PK problem” while skiing.
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

\[ f^u \quad M \quad f^t \]

- \( f \) 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 ≈ AD 50)
Seder

Seder dinner April 1977 at home of Anni Bruss.

“In vino veritas” (Pliny ∼ 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 $\text{gcd}(e, \phi(n)) = 1$
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:
  \[
  C = PK(M) = M^e \mod n \\
  M = SK(C) = C^d \mod n
  \]
Described public-key and RSA cryptosystem in his Scientific American column, *Mathematical Games*
Described public-key and RSA cryptosystem in his Scientific American column, *Mathematical Games*.

Offered copy of RSA technical memo.
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
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 an 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 procedures 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 encrypted form of a message M yields M. Formally,
\[ D(E(M)) = M. \] (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 decipher messages encrypted with E, or compute D efficiently.

(d) If a message M is first deciphered and then enciphered, M is the result. Formally,
\[ D(E(M)) = M. \] (2)

An encryption (or decryption) procedure typically consists of a general method and an encryption key. The general method, under control of the key, encrypts 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 consisting in how much easier it is to decrypt M from C than it is to find M directly. 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 encryption 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 enciphered 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 \( e \cdot d \equiv 1 \pmod{(p - 1) \cdot (q - 1)} \). The security of the system rests in part on the difficulty of factoring the published divisor, n.

E. Diffie, M. Hellman

Key Words and Phrases: digital signatures, public-key cryptosystems, privacy, authentication, security, factorization, prime number, electronic mail, message, transfer, funds transfer, cryptography.

CR Categories: 2.12, 3.15, 3.81, 3.21, 5.25

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)

$PK_A \rightarrow PK_A(M) \leftarrow PK_A$

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:
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

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
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.
MIT started VLSI effort.
MIT started VLSI effort.
R, S, A designed “RSA chip” and fabbed prototype:
RSA on a chip (1980)

- MIT started VLSI effort.
- R, S, A designed “RSA chip” and fabbed prototype:
  - 512-bit bignum processor
RSA on a chip (1980)

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

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:

1. $i = (i + 1) \mod 256$
2. $j = (j + S[i]) \mod 256$
3. swap $S[i]$ and $S[j]$
4. Output $S[(S[i] + S[j]) \mod 256]$

Used in: WEP, BitTorrent, SSL, Kerberos, PDF, Skype, ...

Showing its age (statistical attacks)
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)...


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 $S[(j + S[i]) \mod 256]$

- Used in: WEP, BitTorrent, SSL, Kerberos, PDF, Skype, ...
- Showing its age (statistical attacks)...
Spritz – RC4 replacement (w/ J. Schuldt, 2014)

RC4()

1. $i = i + 1$
2. $j = j + S[i]$
3. $\text{SWAP}(S[i], S[j])$
4. $z = S[S[i] + S[j]]$
5. $\text{return } z$

$\text{SPRITZ}()$

1. $i = i + 1$
2. $j = k + S[j + S[i]]$
3. $k = i + k + S[j]$
4. $\text{SWAP}(S[i], S[j])$
5. $z = S[j + S[i + S[z + k]]]$
6. $\text{return } z$

- 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 $\text{SPRITZ-256}$ 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

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
United States Patent

Rivest et al.

CRYPTOGRAPHIC COMMUNICATIONS SYSTEM AND METHOD


Assignee: Massachusetts Institute of Technology, Cambridge, Mass.

Filed: Dec. 14, 1977

Int. Cl. H04K 1/06; H04L 9/04

Claims 22

Field of Search

References Cited

U.S. Patent Documents

Primary Examiner—Sal Cangialosi

Attorney, Agent, or Firm—Arthur A. Smith, Jr.; Robert J. Henn, Jr.

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 encrypted in 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
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 Dynamics in 1996, now part of EMC.
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

Some pre-1976 context
Invention of Public-Key Crypto and RSA
Early steps
The cryptography business

Crypto policy
Attacks
More New Directions
What Next?

Conclusions
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, corporations, and banks in electronic tunnels. It is easy to send computer programs between connected machines and to instruct a program to search for, select, and report data. The consequences for individuals and for society if computers continue to be connected, as they are now, according to local decisions and informal agreements, are important to understand.

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

Some pre-1976 context
Invention of Public-Key Crypto and RSA
Early steps
The cryptography business
Crypto policy

 Attacks
More New Directions
What Next?
Conclusions
Factorization of RSA-129 (April 1994)

- RSA-129 =

  11438162575788886766923577997614661201021829
  67212423625625618429357069352457338978305971
  23563958705058989075147599290026879543541

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

  34905295108476509491478496199038981334177646
  38493387843990820577 x
  32769132993266709549961988190834461413177642
  967992942539798288533

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

- secret message: The Magic Words Are Squeamish Ossifrage
Factorization of RSA-129 (April 1994)

- RSA-129 =
  11438162575788886766923577997614661201021829
  67212423625625618429357069352457338978305971
  23563958705058989075147599290026879543541

- Derek Atkins, Michael Graff, Arjen Lenstra, Paul Leyland: RSA-129 =
  34905295108476509491478496199038981334177646
  38493387843990820577 x
  32769132993266709549961988190834461413177642
  967992942539798288533

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

secret message: The Magic Words Are Squeamish Ossifrage
Factorization of RSA-129 (April 1994)

- RSA-129 =
  11438162575788886766923577997614661201021829
  67212423625625618429357069352457338978305971
  23563958705058989075147599290026879543541

- Derek Atkins, Michael Graff, Arjen Lenstra, Paul Leyland: RSA-129 =
  34905295108476509491478496199038981334177646
  38493387843990820577 x
  32769132993266709549961988190834461413177642
  967992942539798288533

- 8 months work by about 600 volunteers from more than 20 countries; 5000 MIPS-years.
Factorization of RSA-129 (April 1994)

- **RSA-129 =**
  
  \[
  11438162575788886766923577997614661201021829 \\
  67212423625625618429357069352457338978305971 \\
  23563958705058989075147599290026879543541 
  \]

- Derek Atkins, Michael Graff, Arjen Lenstra, Paul Leyland: **RSA-129 =**
  
  \[
  34905295108476509491478496199038981334177646 \\
  38493387843990820577 \times \\
  32769132993266709549961988190834461413177642 \\
  967992942539798288533 
  \]

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

- secret message:
  
  The Magic Words Are Squeamish Ossifrage
010, 218, 296, 72

For Solving the Scientific American RSA Challenge

BayBank
Massachusetts

Official Bank Check

Date April 22, 1994

To the order of:

**Derek Atkins or Michael Graff or Arjen Lenstra or Paul Leyland**

$100,000

AMOUNTS IN EXCESS OF $100,000.00 REQUIRE TWO SIGNATURES

Authorized Signature

Authorized Signature
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 $n$. 

\[ |\psi\rangle = \alpha |0\rangle + \beta |1\rangle \]
\[ |\alpha|^2 + |\beta|^2 = 1 \]
\[ \alpha |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \beta |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \]
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 $15 = 3 \times 5$. 

$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$

$|\alpha|^2 + |\beta|^2 = 1$

$\alpha |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$\beta |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
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): $291311 = 557 \times 523$
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): $291311 = 557 \times 523$
- Dark clouds on horizon for RSA?
In 2004 Xiaoyun Wang and colleagues found a way to produce collisions for MD5:

\[ \text{MD}^5(\text{file1}) = \text{MD}^5(\text{file2}) \]

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:

\[ \text{MD}^5(file1) = \text{MD}^5(file2) \]

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

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
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

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!
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!
Zero-Knowledge Proofs

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

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!
Probabilistic MicroPayment System

- Micali and Rivest (2001)

- Paying ten cents ≡ paying $1 with probability 1/10.

- Uses pseudorandom digital signatures for "verifiable fair dice."

Probabilistic MicroPayment System

- Micali and Rivest (2001)
- Paying ten cents \(\equiv\) paying $1 with probability 1/10. Uses pseudorandom digital signatures for “verifiable fair dice.”
Probabilistic MicroPayment System

- Micali and Rivest (2001)
- Paying ten cents $\equiv$ paying $1$ with probability $1/10$. Uses pseudorandom digital signatures for “verifiable fair dice.”
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!
In 1978, Rivest, Adleman, and Dertouzos asked, “Can one compute on encrypted data, while keeping it encrypted?”
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

Some pre-1976 context

Invention of Public-Key Crypto and RSA

Early steps

The cryptography business

Crypto policy

Attacks

More New Directions

What Next?

Conclusions
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!