The death and rebirth of classical cryptography in a quantum world

Goutam Paul
http://www.goutampaul.com

Cryptology and Security Research Unit,
Indian Statistical Institute, Kolkata

February 10, 2016

Lecture at
International School and Conference on Quantum Information,
Institute of Physics (IOP), Bhubaneswar (Feb 9-18, 2016).

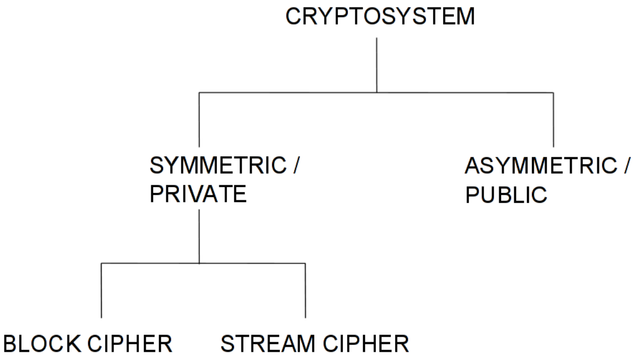
Outline

1. Pre-Quantum Cryptograpghy
   - Public Key Cryptography
   - RSA
2. Quantum Attacks on Classical Cryptosystems
   - Solving Hard Problems by Quantum Computers
   - Death of Classical Public Key Cryptography
   - Need for QKD
3. Quantum Cryptography
   - Quantum Key Distribution (QKD)
   - Other Quantum Cryptography Algorithms
4. Post-Quantum Cryptography
   - Rebirth of Classical Cryptography

Roadmap

1. Pre-Quantum Cryptograpghy
   - Public Key Cryptography
   - RSA
2. Quantum Attacks on Classical Cryptosystems
   - Solving Hard Problems by Quantum Computers
   - Death of Classical Public Key Cryptography
   - Need for QKD
3. Quantum Cryptography
   - Quantum Key Distribution (QKD)
   - Other Quantum Cryptography Algorithms
4. Post-Quantum Cryptography
   - Rebirth of Classical Cryptography

The Crypto World



PKC: Origin and History

Timeline

- 1976: The Idea - Whitfield Diffie and Martin Hellman
- 1976: Diffie and Hellman Key Exchange algorithm
- 1978: Rivest, Shamir and Adleman invented RSA

Actual Timeline (?) [announced in 1997]

- 1970: The Idea - James H. Ellis (British intelligence)
- 1973: Clifford Cocks developed RSA algorithm
- 1974: Malcom Williamson built Diffie-Hellman scheme

Public Key Framework

Goal: Alice and Bob communicate securely, avoiding Charles

Alice (receiver)	KEY GEN: Construct *related pair* of keys (public and private)
	KEY DIST: Publish public key and keep private key secret
Bob (sender)	GET KEY: Obtain an authentic Public Key of Alice
	ENCRYPT: Use it to encrypt message and send to Alice
Alice (receiver)	GET CIPHER: Obtain the ciphertext sent by Bob
	DECRYPT: Use Private Key to decrypt the ciphertext

Examples of Public Key Cryptosystems

Examples of Public Key Cryptosystems

- RSA (1977)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)
- Goldwasser-Micali (1982)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)
- Goldwasser-Micali (1982)
- ElGamal (1985)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)
- Goldwasser-Micali (1982)
- ElGamal (1985)
- ECC (1985)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)
- Goldwasser-Micali (1982)
- ElGamal (1985)
- ECC (1985)
- Cramer-Shoup (1998)

Examples of Public Key Cryptosystems

- RSA (1977)
- Knapsack (1978)
- Goldwasser-Micali (1982)
- ElGamal (1985)
- ECC (1985)
- Cramer-Shoup (1998)
- Paillier (1999)

Example: RSA Cryptosystem

Key Gen	Choose two *large* primes p and q
	Compute the product $N = pq$
	Compute Euler's Totient function $\phi(N) = (p-1)(q-1)$
	Choose positive integer e such that $\gcd(e, \phi(N)) = 1$
	Compute d such that $ed \equiv 1 \pmod{\phi(N)}$
Key Dist	Public Key $= \langle N, e \rangle$ and Private Key $= \langle N, d \rangle$
Encryption	Message M produces Ciphertext $C = M^e \bmod N$
Decryption	Ciphertext C produces Message $M = C^d \bmod N$

Example: an RSA Instance

Example: an RSA Instance

- Suppose $p = 653, q = 877$. Then $N = pq = 572681$, $\phi(N) = (p-1)(q-1) = 571152$.

Example: an RSA Instance

- Suppose $p = 653, q = 877$. Then $N = pq = 572681$, $\phi(N) = (p-1)(q-1) = 571152$.
- Suppose Bob chooses $e = 13$ as the encryption exponent.

Example: an RSA Instance

- Suppose $p = 653, q = 877$. Then $N = pq = 572681$, $\phi(N) = (p-1)(q-1) = 571152$.
- Suppose Bob chooses $e = 13$ as the encryption exponent.
- Now he has to find the decryption exponent d which is e^{-1} in $\mathbb{Z}_{\phi(N)}$.

Example: an RSA Instance

- Suppose $p = 653, q = 877$. Then $N = pq = 572681$, $\phi(N) = (p-1)(q-1) = 571152$.
- Suppose Bob chooses $e = 13$ as the encryption exponent.
- Now he has to find the decryption exponent d which is e^{-1} in $\mathbb{Z}_{\phi(N)}$.
- One can check that $13 \times 395413 \equiv 1 \pmod{571152}$.

Example: an RSA Instance

- Suppose $p = 653, q = 877$. Then $N = pq = 572681$, $\phi(N) = (p-1)(q-1) = 571152$.
- Suppose Bob chooses $e = 13$ as the encryption exponent.
- Now he has to find the decryption exponent d which is e^{-1} in $\mathbb{Z}_{\phi(N)}$.
- One can check that $13 \times 395413 \equiv 1 \pmod{571152}$.
- Hence, the RSA parameters for Bob are
    - public key: $(13, 572681)$, and
    - private key: $(395413, 572681)$.

Example: an RSA Instance (contd...)

Example: an RSA Instance (contd...)

- To encrypt a plaintext $m = 12345$, Alice uses Bob's public key $(13, 572681)$, and calculates $c = 12345^{13} \bmod 572681 = 536754$ and sends c to Bob.

Example: an RSA Instance (contd...)

- To encrypt a plaintext $m = 12345$, Alice uses Bob's public key $(13, 572681)$, and calculates $c = 12345^{13} \bmod 572681 = 536754$ and sends c to Bob.
- To decrypt $c = 536754$, Bob calculates $536754^{395413} \bmod 572681 = 12345 = m$.

Correctness and Security of RSA

Correctness depends on EULER FERMAT theorem

$$a^{\phi(n)} \equiv 1 \pmod{n} \quad \text{if} \quad \gcd(n, a) = 1$$

Security depends on FACTORIZATION problem

Obtain factors p, q given product $N = pq$

Factoring Challenge

What	Digits	Who	When
7141075053842	13	Carissan (Machine à Congruences)	1919
9999000099990001	16	Lehmer (Bicycle Sieve)	1926
$2^{93}+1$	28	Lehmer (Gear Sieve)	1932
RSA-129	129	600 volunteers all over the world (MPQS)	1994
RSA-130	130	Lenstra and group (GNFS)	1996
RSA-140	140	Montgomery, Leyland, Dodson, Zimmermann, Lenstra (GNFS)	1999
RSA-155	155	Muffet, Leyland, Montgomery, Dodson, Morain, Guillerm,Marchand, Lenstra, Zimmermann, Gilchrist, Aardal, Putnam (GNFS)	1999
$2^{953}+1$	158	Bahr, Boehm, Franke, Kleinjung (GNFS)	2002
RSA-160	160	Bundesamt für Sicherheit in der Informationstechnik (BSI) Researchers (GNFS)	2002
RSA-576	174	Franke, Kleinjung, Montgomery, te Riele, Bahr, Leclair, Leyland, Wackerbarth (GNFS)	2003
$11^{281}+1$	176	Aoki, Kida, Shimoyama and Ueda (GNFS)	2005
RSA-640	193	Bahr, Boehm, Franke, Kleinjung (GNFS)	2005
RSA-200	200	Bahr, Boehm, Franke, Kleinjung (GNFS)	2005

Best: RSA-768 (232 digits) factored by several researchers in 2010 (over 2 years)

Roadmap

1. Pre-Quantum Cryptograpghy
   - Public Key Cryptography
   - RSA
2. Quantum Attacks on Classical Cryptosystems
   - Solving Hard Problems by Quantum Computers
   - Death of Classical Public Key Cryptography
   - Need for QKD
3. Quantum Cryptography
   - Quantum Key Distribution (QKD)
   - Other Quantum Cryptography Algorithms
4. Post-Quantum Cryptography
   - Rebirth of Classical Cryptography

Period finding problem

Let

$$f : \{0, 1, 2, \ldots, M-1\} \to \{0, 1, 2, \ldots, M-1\}$$

be a periodic function of period r, meaning that

$$f(x) = f(x+r) \quad \forall x \in \{0, 1, 2, \ldots, M-1\}$$

and the values $f(x), f(x+1), f(x+2), \ldots, f(x+r-1)$ are all distinct.

For simplicity, one can assume that $M = 2^m$ that $r \leq M/2$.
Finding the unknown period is a hard problem in classical computing.

Quantum Algorithm for finding period

4. **Create the quantum state** $\frac{1}{\sqrt{M}} \sum_x |x\rangle |f(x)\rangle$.
5. **Measure the last** m **bits of the state**: for an output $y = f(x_0)$ with the smallest possible x_0, the residual state is

$$\frac{1}{\sqrt{[\frac{M}{r}]}} \sum_{t=0}^{[\frac{M}{r}]-1} |x_0 + tr\rangle |f(x_0)\rangle.$$

6. Ignore the last n bits and apply the Fourier transform to the first m bits to get

$$\frac{1}{\sqrt{M}} \frac{1}{\sqrt{[\frac{M}{r}]}} \sum_s \sum_{t=0}^{[\frac{M}{r}]-1} \omega^{(x_0+tr)\cdot s} |s\rangle.$$

Quantum Algorithm for finding period (contd...)

1. Measurement gives an integer s with probability

$$\frac{1}{M} \cdot \frac{1}{[\frac{M}{r}]} |\omega^{x_0 s}|^2 \left| \sum_{t=0}^{[\frac{M}{r}]-1} \omega^{(x_0+tr)\cdot s} \right|^2 = \frac{1}{M} \cdot \frac{1}{[\frac{M}{r}]} \left| \sum_{t=0}^{[\frac{M}{r}]-1} \omega^{trs} \right|^2 .$$

2. This probability is higher, the closer the unit vector ω^{rs} is to the positive real axis, or the closer rs/M is to some integer c.
3. Known value $s/M \approx$ unknown value c/r.
   This information suffices to determine r.

Order finding problem

For $a \in \mathbb{Z}_N^*$, the order of $a \in \mathbb{Z}_N^*$ (or the order of a modulo N) is the *smallest* positive integer r such that

$$a^r \equiv 1(\mod N).$$

The order finding problem is to find the order of an element a, given an integer $N \geq 2$ and an element $a \in \mathbb{Z}_N^*$.

Classically this problem is hard. But, quantum period finding can be used to solve order finding.

Reducing factoring to order finding

- Suppose that the random choice of a is in $\mathbb{Z}_N^*$ (which is very likely), and that the order r of a is even.
- N divides $a^r - 1 = (a^{r/2} + 1)(a^{r/2} - 1)$.
- N cannot divide $a^{r/2} - 1$, otherwise $r/2 < r$ would have been the order.
- If $N \nmid a^{r/2} + 1$ (lucky case), $gcd(N, a^{r/2} - 1)$ gives a non-trivial factor of N.

Efficiency of Shor's Algorithm, 1994

Efficiency of Shor's Algorithm, 1994

- Fastest classical algorithm has sub-exponential time complexity: $O(e^{1.9(\log N)^{1/3}(\log \log N)^{2/3}})$.

Efficiency of Shor's Algorithm, 1994

- Fastest classical algorithm has sub-exponential time complexity: $O(e^{1.9(\log N)^{1/3}(\log\log N)^{2/3}})$.
- Polynomial time quantum algorithm known due to Shor, 1994: $O\left((\log N)^2(\log\log N)(\log\log\log N)\right)$.

Efficiency of Shor's Algorithm, 1994

- Fastest classical algorithm has sub-exponential time complexity: $O(e^{1.9(\log N)^{1/3}(\log\log N)^{2/3}})$.
- Polynomial time quantum algorithm known due to Shor, 1994: $O\left((\log N)^2(\log\log N)(\log\log\log N)\right)$.
- In 2001, a group at IBM factored 15, using an NMR quantum computer with 7 qubits.

Efficiency of Shor's Algorithm, 1994

- Fastest classical algorithm has sub-exponential time complexity: $O(e^{1.9(\log N)^{1/3}(\log\log N)^{2/3}})$.
- Polynomial time quantum algorithm known due to Shor, 1994: $O\left((\log N)^2(\log\log N)(\log\log\log N)\right)$.
- In 2001, a group at IBM factored 15, using an NMR quantum computer with 7 qubits.
- Until 2012 the largest number factored using Shor's algorithm was 15.

Efficiency of Shor's Algorithm, 1994

- Fastest classical algorithm has sub-exponential time complexity: $O(e^{1.9(\log N)^{1/3}(\log\log N)^{2/3}})$.
- Polynomial time quantum algorithm known due to Shor, 1994: $O\left((\log N)^2(\log\log N)(\log\log\log N)\right)$.
- In 2001, a group at IBM factored 15, using an NMR quantum computer with 7 qubits.
- Until 2012 the largest number factored using Shor's algorithm was 15.
- So far, the largest number factored by a quantum computer is 56153, using 4 qubits in an NMR system (Chinese group, PRL 2012).

Implication of Shor's Algorithm

Implication of Shor's Algorithm

- Factoring – breaks RSA (banking, online shopping dead).

Implication of Shor's Algorithm

- Factoring – breaks RSA (banking, online shopping dead).
- Factoring can be used to easily solve
  - quadratic residuosity problem – breaks Goldwasser-Micali.

Implication of Shor's Algorithm

- Factoring – breaks RSA (banking, online shopping dead).
- Factoring can be used to easily solve
  - quadratic residuosity problem – breaks Goldwasser-Micali.
  - decisional composite residuosity problem – breaks Paillier.

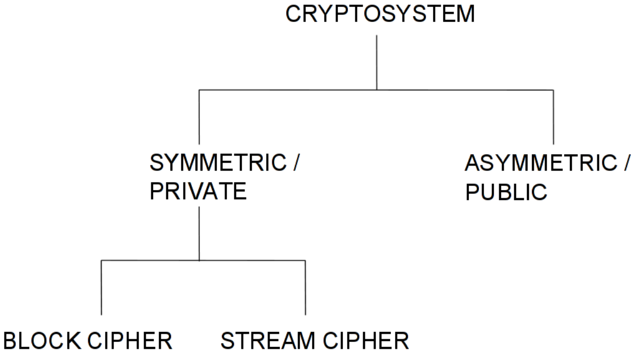
Implication of Shor's Algorithm

- Factoring – breaks RSA (banking, online shopping dead).
- Factoring can be used to easily solve
  - quadratic residuosity problem – breaks Goldwasser-Micali.
  - decisional composite residuosity problem – breaks Paillier.
- Discrete Log – breaks ElGamal, ECC, Cramer-Shoup.

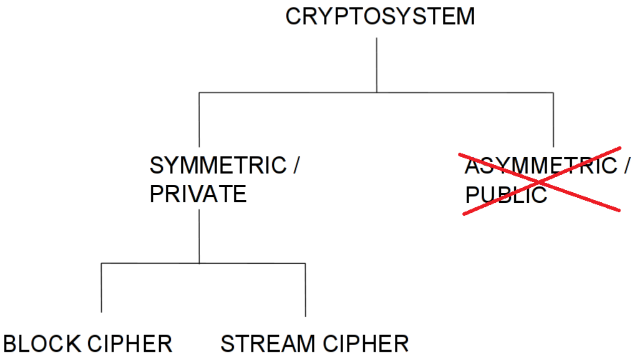
Implication of Shor's Algorithm

- Factoring – breaks RSA (banking, online shopping dead).
- Factoring can be used to easily solve
  - quadratic residuosity problem – breaks Goldwasser-Micali.
  - decisional composite residuosity problem – breaks Paillier.
- Discrete Log – breaks ElGamal, ECC, Cramer-Shoup.
- Shor's algorithm for discrete logarithm can be generalized to find hidden subgroups in abelian groups.

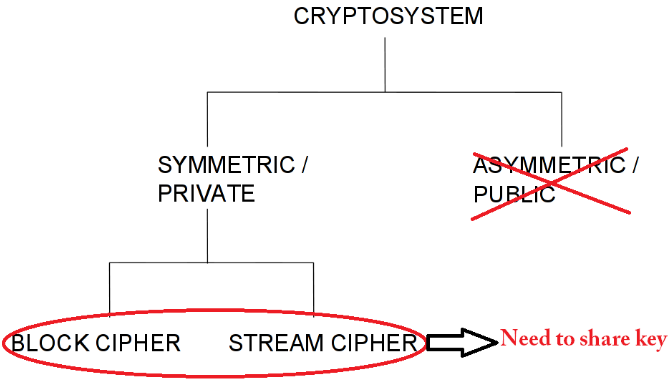
Need for Quantum Key Distribution



Need for Quantum Key Distribution



Need for Quantum Key Distribution



Roadmap

1. Pre-Quantum Cryptograpghy
   - Public Key Cryptography
   - RSA
2. Quantum Attacks on Classical Cryptosystems
   - Solving Hard Problems by Quantum Computers
   - Death of Classical Public Key Cryptography
   - Need for QKD
3. Quantum Cryptography
   - Quantum Key Distribution (QKD)
   - Other Quantum Cryptography Algorithms
4. Post-Quantum Cryptography
   - Rebirth of Classical Cryptography

BB84 Protocol

Uses two conjugate bases $+ = \{\uparrow, \rightarrow\}$ and $\times = \{\nearrow, \nwarrow\}$ to establish a secret key between two parties at a distance.

Alice's bit	0	1	1	0	1	0	0	1
Alice's basis	+	+	X	+	X	X	X	+
Alice's polarization	$\uparrow$	$\rightarrow$	$\nwarrow$	$\uparrow$	$\nwarrow$	$\nearrow$	$\nearrow$	$\rightarrow$
Bob's basis	+	X	X	X	+	X	+	+
Bob's measurement	$\uparrow$	$\nearrow$	$\nwarrow$	$\nearrow$	$\rightarrow$	$\nearrow$	$\rightarrow$	$\rightarrow$
Public discussion								
Shared Secret key	0		1			0		1

Other variants of QKD

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]
- Device Independent (DI) QKD

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]
- Device Independent (DI) QKD
  - Idea by Mayers and Yao [FOCS, 1998]

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]
- Device Independent (DI) QKD
  - Idea by Mayers and Yao [FOCS, 1998]
  - Measurement Device Independent (MDI) QKD [Lo, Curty and Qi, PRL, 2012]

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]
- Device Independent (DI) QKD
  - Idea by Mayers and Yao [FOCS, 1998]
  - Measurement Device Independent (MDI) QKD [Lo, Curty and Qi, PRL, 2012]
  - Side Channel Free (SCF) QKD [Braunstein and Pirandola, PRL, 2012]

Other variants of QKD

- E91 Protocol [Ekert, PRL 1991]
- Semi-Quantum QKD [Boyer, Kenigsberg and Mor, PRL 2007]
- Device Independent (DI) QKD
  - Idea by Mayers and Yao [FOCS, 1998]
  - Measurement Device Independent (MDI) QKD [Lo, Curty and Qi, PRL, 2012]
  - Side Channel Free (SCF) QKD [Braunstein and Pirandola, PRL, 2012]
  - Fully Device Independent (FDI) QKD [Vazirani and Vidick, PRL, 2014]

Non-QKD Quantum Crypto

- Quantum commitment
- Quantum SMC
- Position-based quantum cryptography

Roadmap

1. Pre-Quantum Cryptograpghy
   - Public Key Cryptography
   - RSA
2. Quantum Attacks on Classical Cryptosystems
   - Solving Hard Problems by Quantum Computers
   - Death of Classical Public Key Cryptography
   - Need for QKD
3. Quantum Cryptography
   - Quantum Key Distribution (QKD)
   - Other Quantum Cryptography Algorithms
4. Post-Quantum Cryptography
   - Rebirth of Classical Cryptography

Post-Quantum Cryptography

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)
- Multivariate cryptography (e.g., Rainbow)

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)
- Multivariate cryptography (e.g., Rainbow)
- Hash-based cryptography (e.g., Lamport, Merkle).

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)
- Multivariate cryptography (e.g., Rainbow)
- Hash-based cryptography (e.g., Lamport, Merkle).
- Code-based cryptography (e.g., McEliece, Niederreiter)

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)
- Multivariate cryptography (e.g., Rainbow)
- Hash-based cryptography (e.g., Lamport, Merkle).
- Code-based cryptography (e.g., McEliece, Niederreiter)
- Supersingular ECC

Post-Quantum Cryptography

- Lattice-based cryptography (e.g., NTRU)
- Multivariate cryptography (e.g., Rainbow)
- Hash-based cryptography (e.g., Lamport, Merkle).
- Code-based cryptography (e.g., McEliece, Niederreiter)
- Supersingular ECC
- Symmetric Key Cryptography

Thank You

Questions / Comments ?

Homepage: http://www.goutampaul.com
Email: goutam.k.paul@gmail.com