

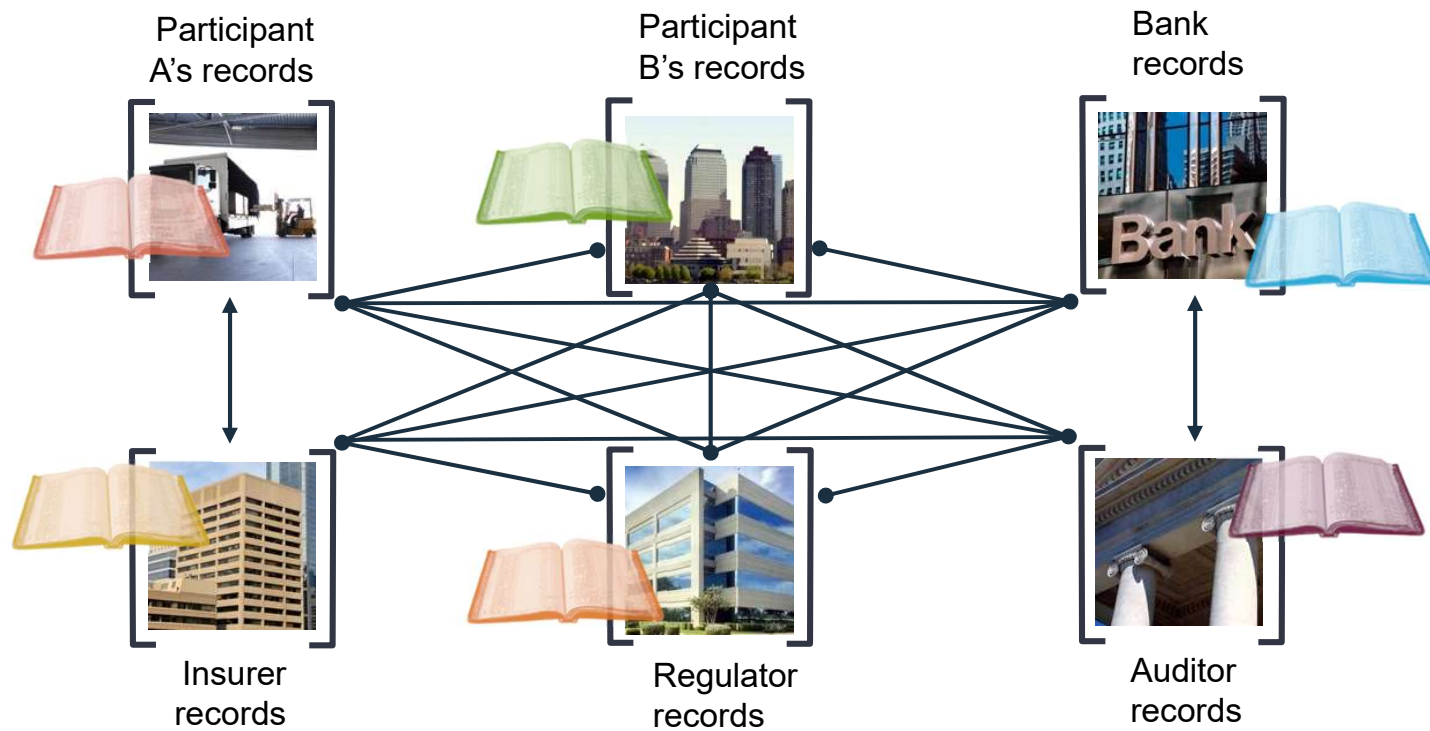
Introduction to the Cryptography Behind Blockchain (from roots to Quantum)

Marcelo Sávio

IBM Industry Solutions Architect

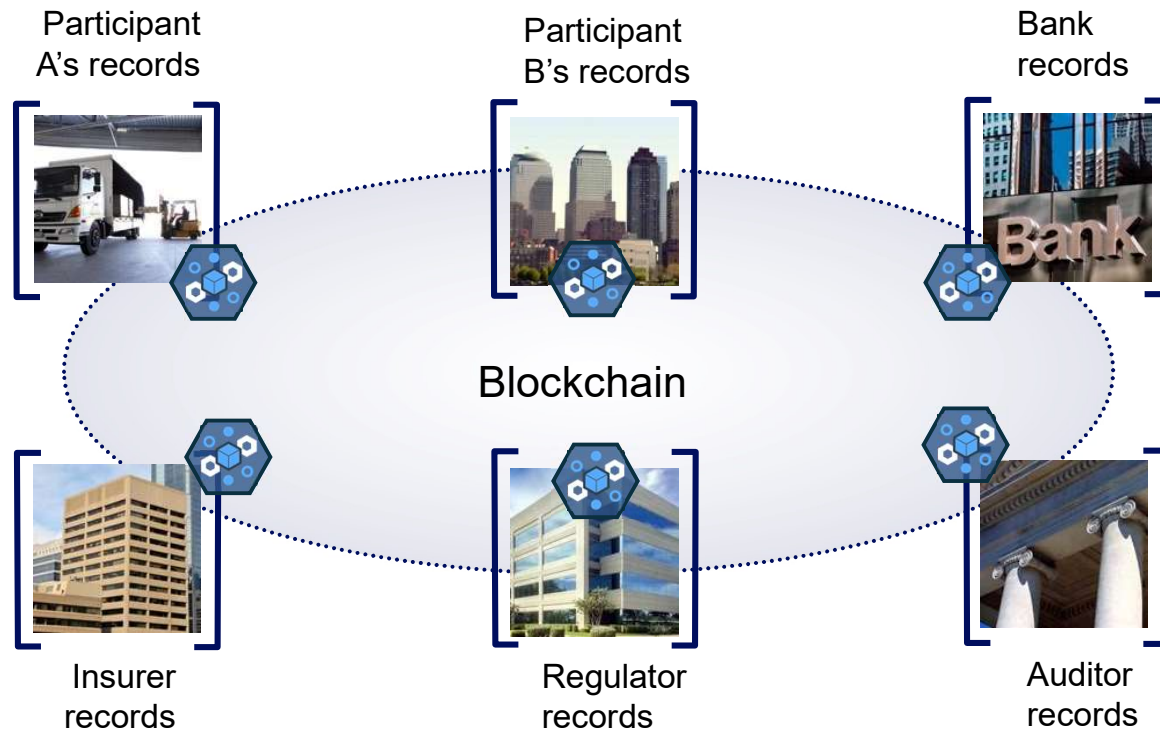
<https://www.linkedin.com/in/msavio/>

The Problem...



... inefficient, expensive, vulnerable

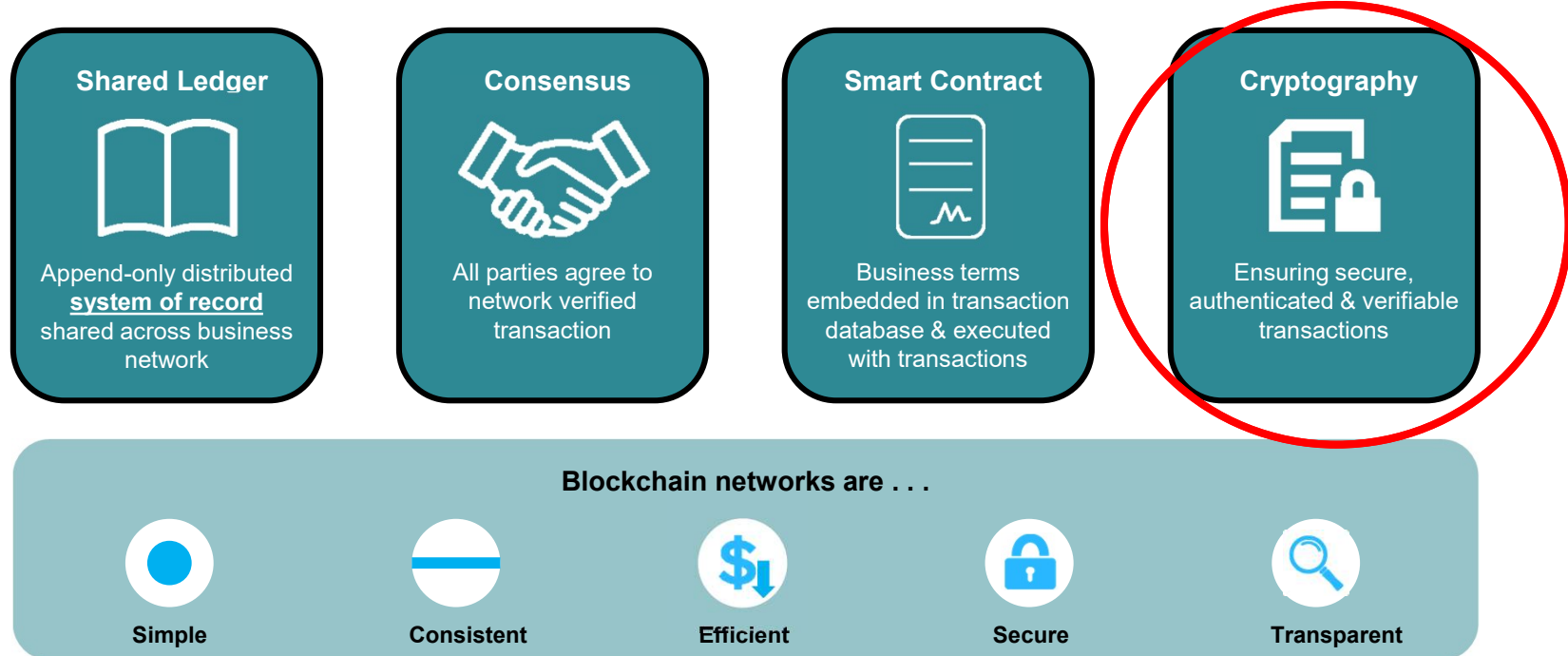
The Solution: A shared, replicated, permissioned ledger ...



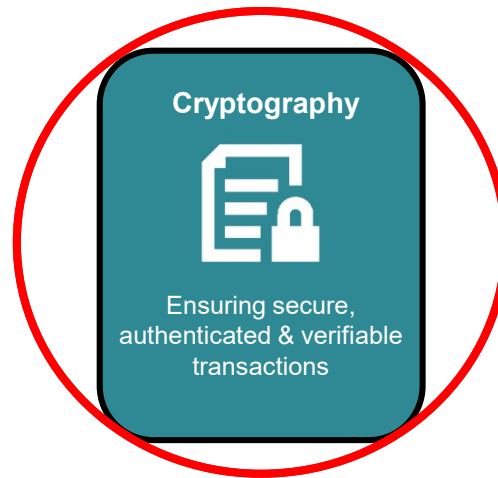
... with consensus, provenance, immutability and finality

How does it work?

Blockchain is a technology platform that creates a **distributed ledger** between participants to allow transfer of asset ownership or transactions using “**smart contracts**” in a secured and permissioned environment guaranteed by **cryptographic** techniques and algorithms.



Where else does Blockchain cryptography work?



All the cryptographic techniques and algorithms used in Blockchain have been around for quite a long time, being used on the Internet (e-commerce, e-banking and many other applications), cell phones etc.

BLOCKCHAIN
(Electronic Ledger)

BITCOIN
(Crypto Currency)



BLOCKCHAIN
(Electronic Ledger)

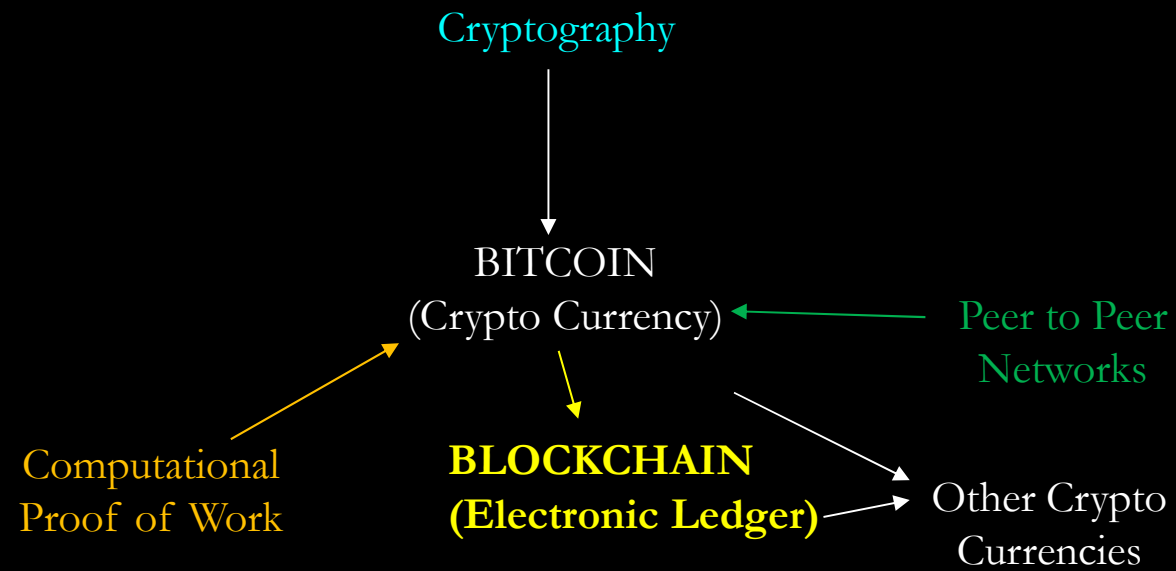
BITCOIN
(Crypto Currency)

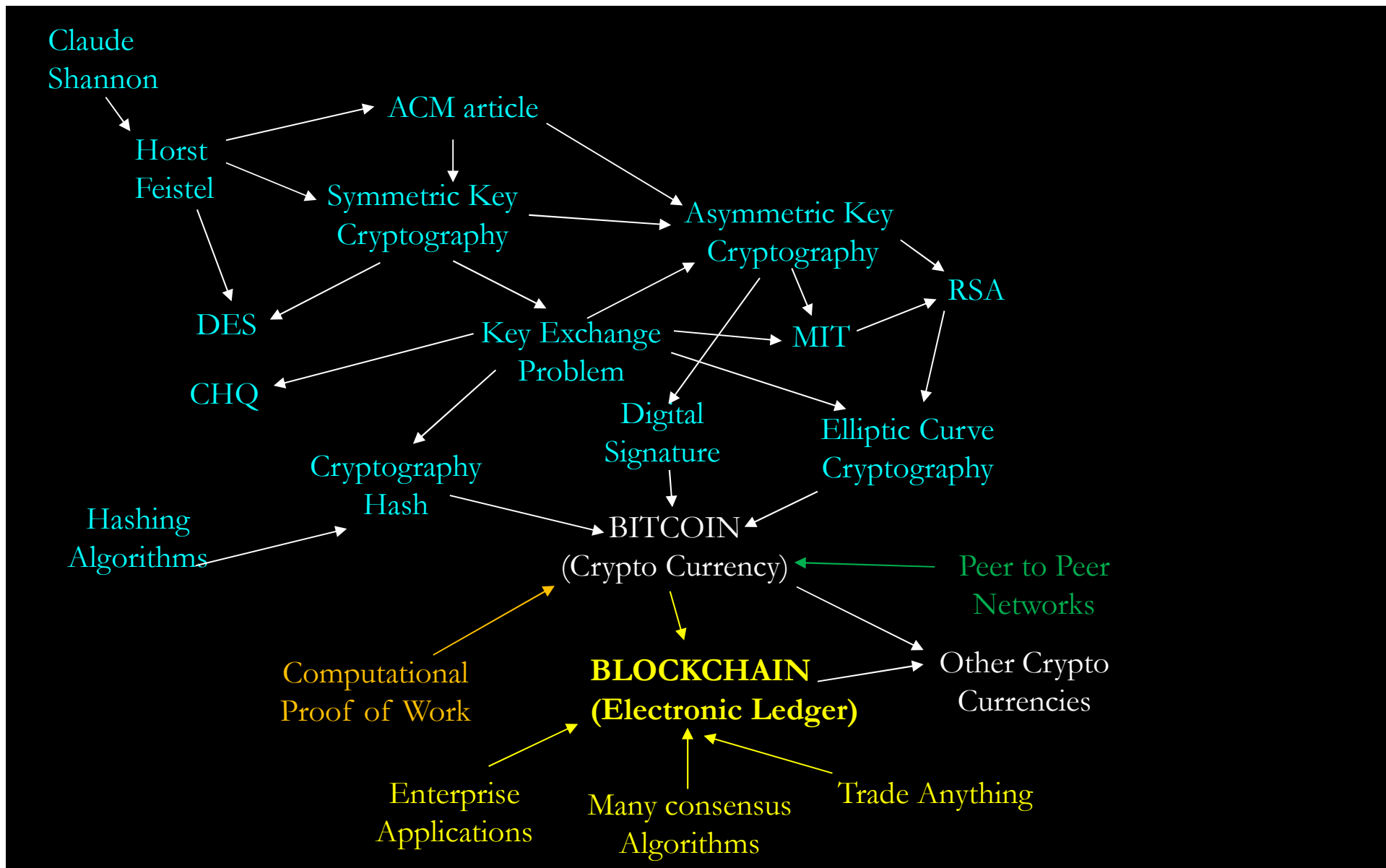


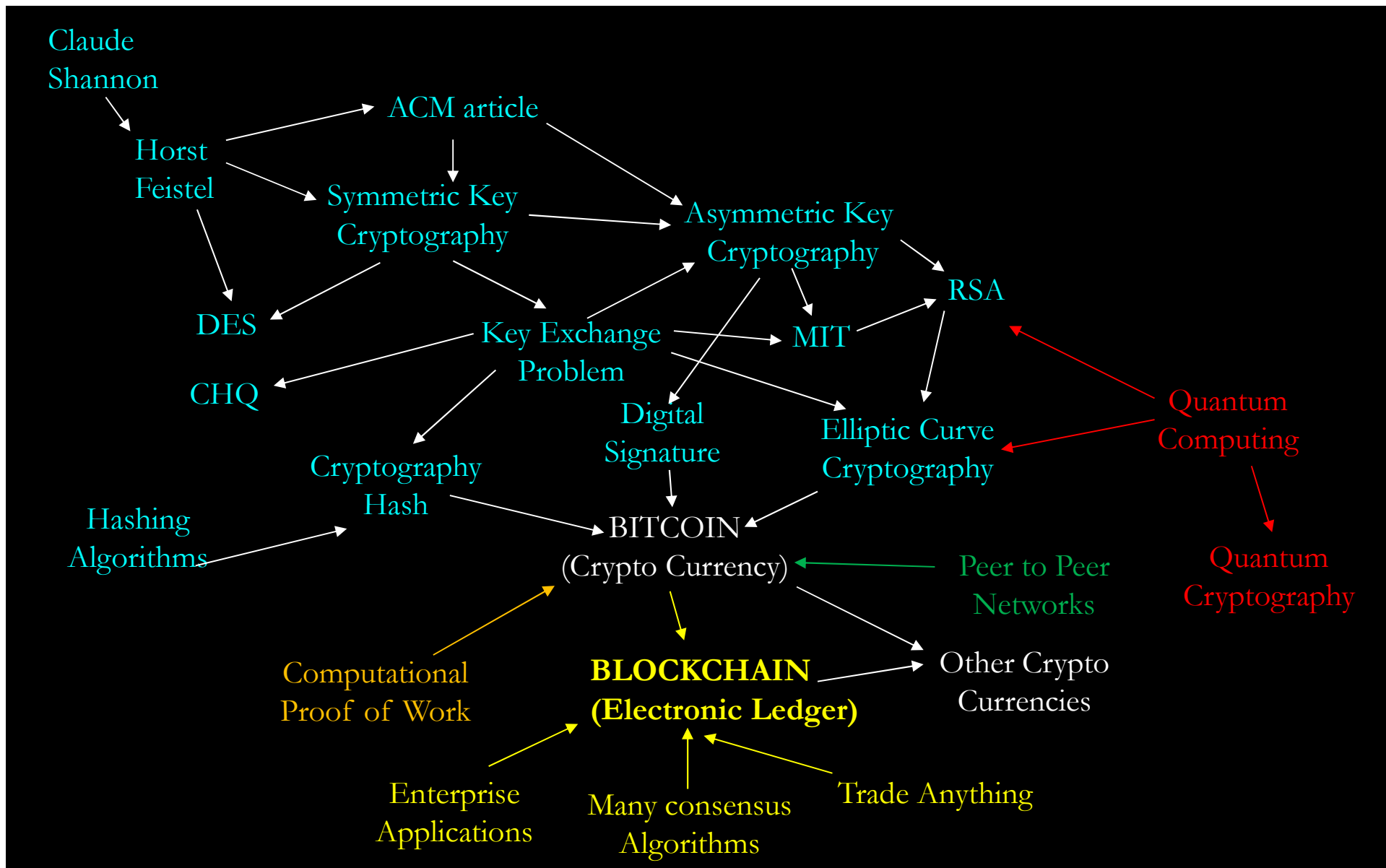
BLOCKCHAIN
(Electronic Ledger)



Other Crypto
Currencies









Cryptography: A Short History

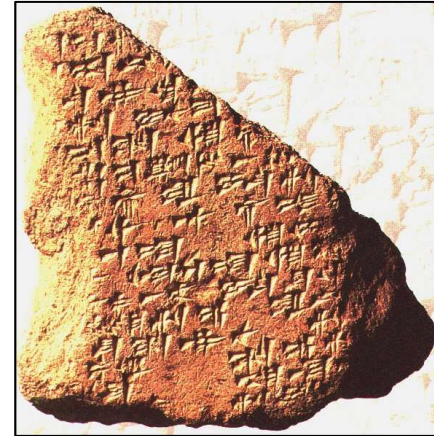
I am NOT going to talk about this...



Ancient Egypt



Middle Ages

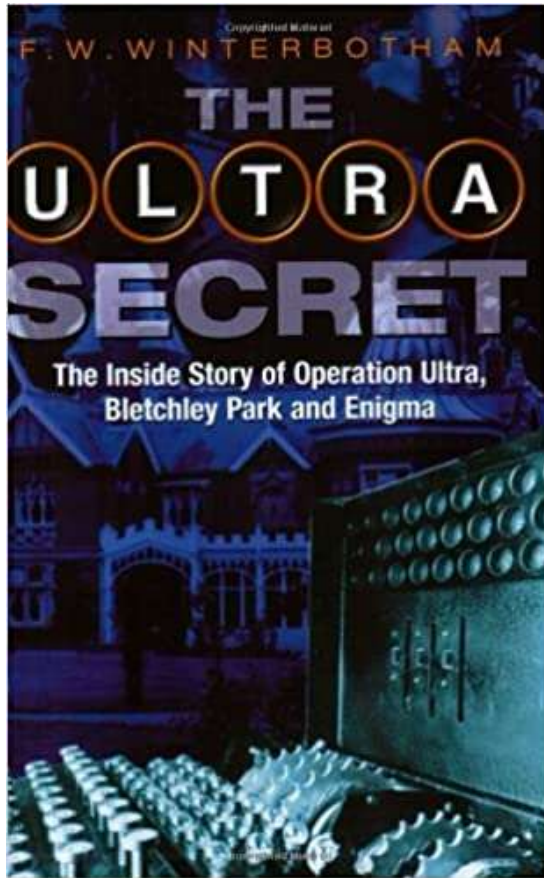


Ancient Mesopotamia



Ancient Greece

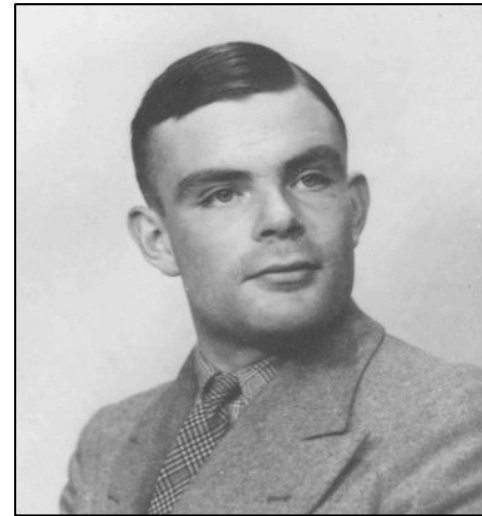
Nor this...



Bletchley Park, UK



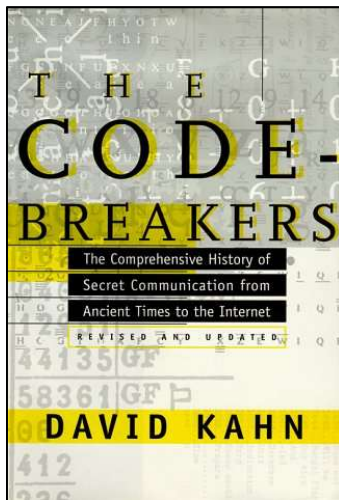
Enigma



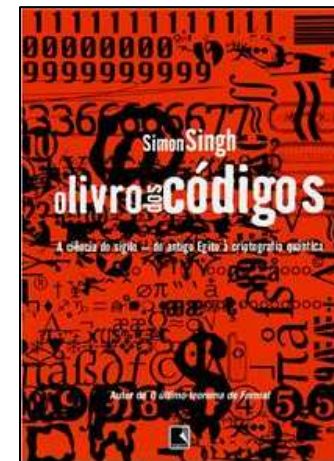
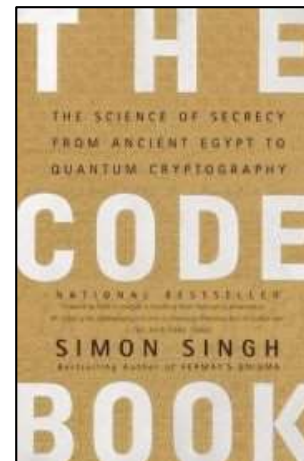
Alan Turing (1912-1954)

But you definitely should learn about this fascinating history.

And for that I recommend these two books:

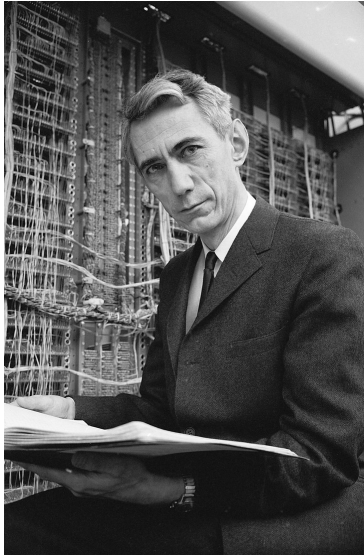


- ***The Codebreakers***
by David Kahn,
1996, Scribner



- ***The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography***,
by Simon Singh, 1999, Anchor Books
(Available in Brazilian Portuguese)

Our talk today starts in the middle of the last Century...

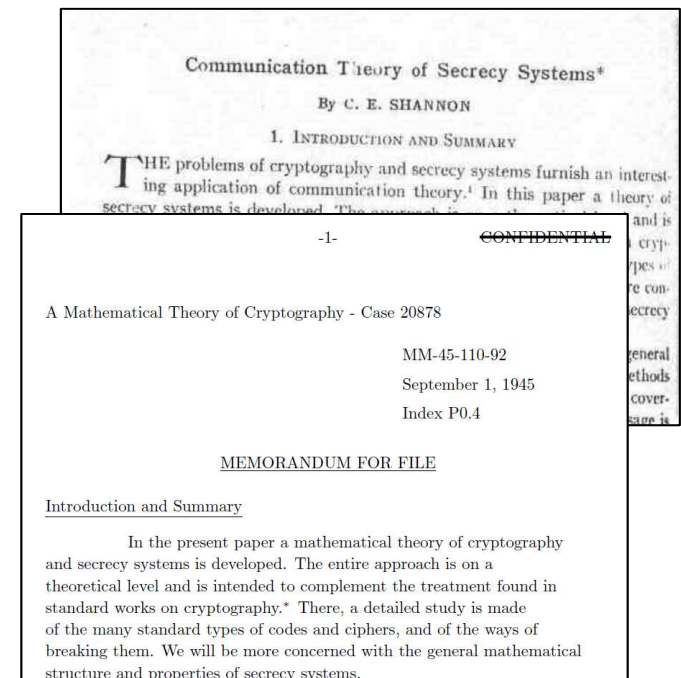
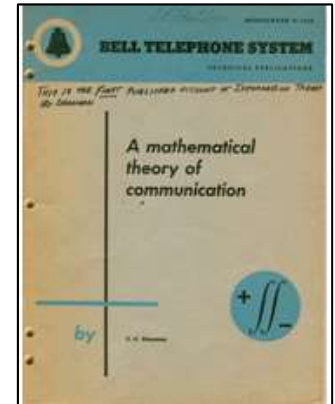


Claude Elwood Shannon (1916 - 2001)

Joined AT&T Bell Telephones (New Jersey, USA) in 1941 as a research mathematician and remained there until 1972. Published "*A Mathematical Theory of Communication*" in 1948 which founded the Information Theory.

In Oct 1949, published another paper entitled "**Communication Theory of Secrecy Systems**", which is generally credited with transforming cryptography from an art to a science.

Actually this paper was a shorter and declassified version of a previous paper Shannon published in Sep 1945 in a Confidential Classified report entitled “***A Mathematical Theory of Cryptography***”. Shannon developed the information theory concepts in support of cryptography.



Cryptography was an almost exclusively military matter, until...



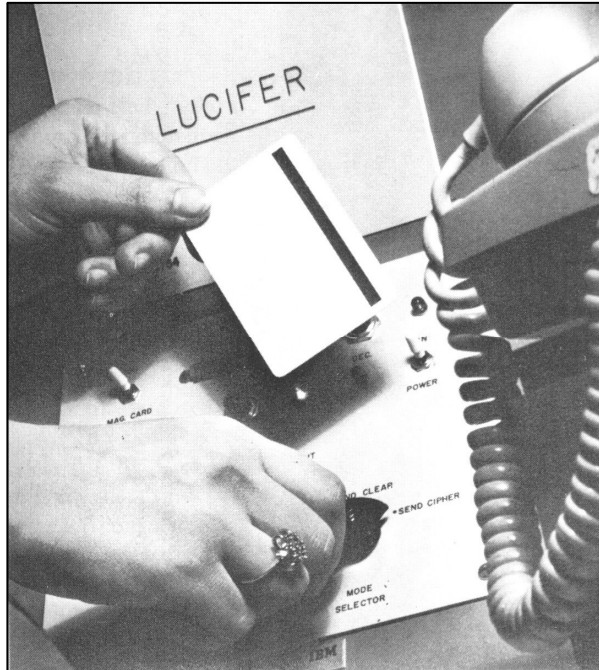
Horst Feistel (1915 - 1990)

Was a German-born cryptographer who immigrated to the US and in 1968 joined IBM T.J. Watson Research Center in Yorktown Heights, NY.

There he led a lab which was the only significant non-governmental cryptographic research group in the US (and probably in the world) at the time.

IBM filed several patents based on his cryptographic ideas built on top of Claude Shannon's concepts (confusion, diffusion etc.)

He joined IBM in a very propitious moment, just in time to solve the problem of security of electronic transactions on Automated Teller Machines for the Lloyds bank cashpoint system.

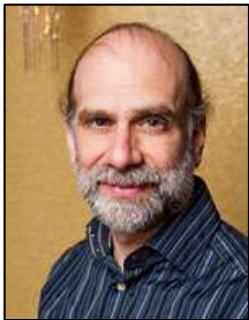


Feistel developed in the APL Programming Language a block cipher cryptographic system called LUCIFER to protect the data for the remote cash-dispensing system.

This not only represented a new paradigm for encryption systems but also mainly brought encryption from a little-known military science into our daily lives, and stimulated research in cryptography and competition in creating encryption algorithms.

The LUCIFER algorithm was then submitted by IBM to the US National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of government data.

In January 1977 it was published as an official Federal Information Processing Standard (FIPS) for the United States, in which it became known as **Data Encryption Standard (DES)** and became widely used – and not only by the US Government - for almost 40 years.

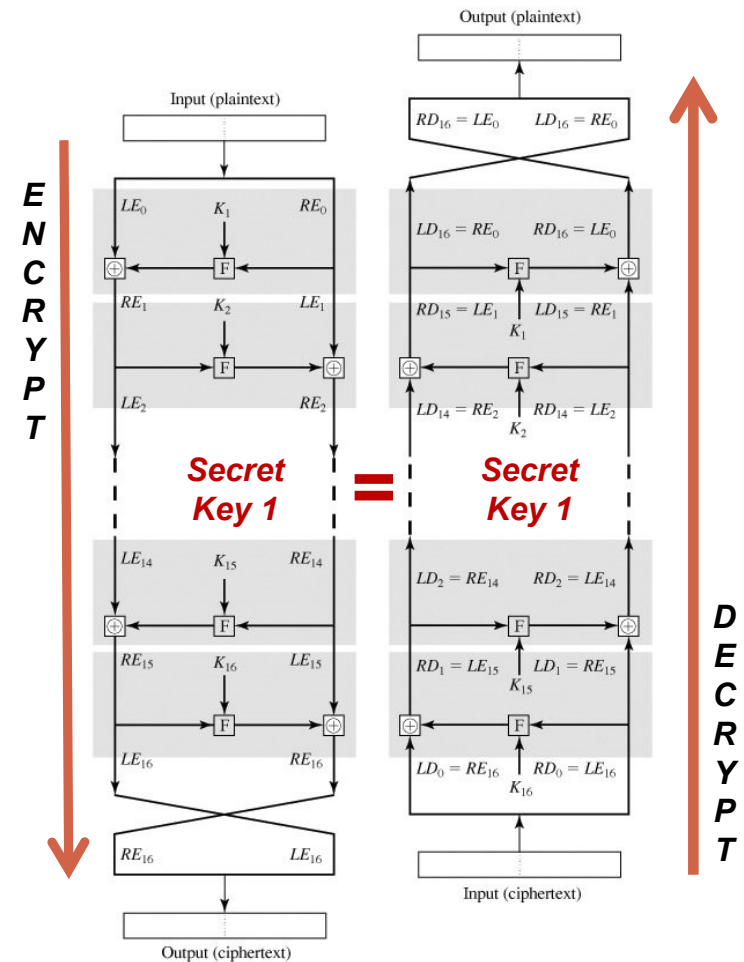


“Almost all of the encryption algorithms ... can trace their roots back to DES.”

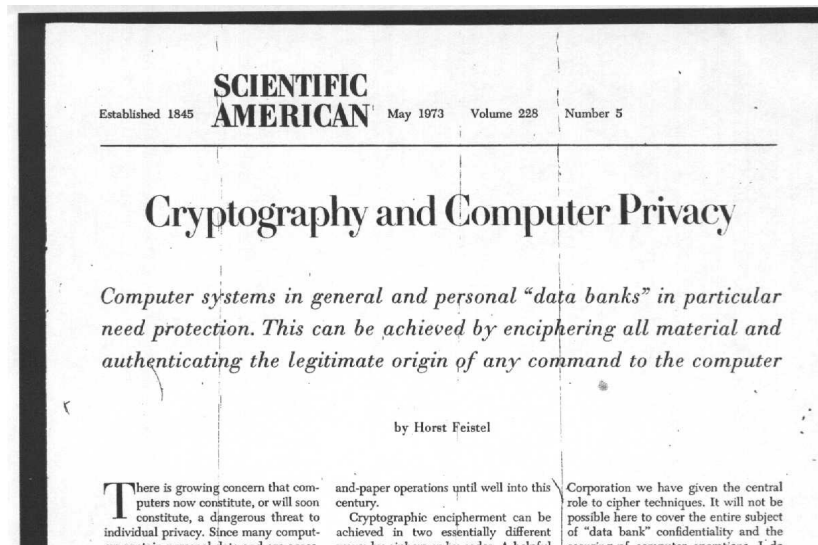
Bruce Schneier, Information Security Expert

<https://www.schneier.com/>

- The Data Encryption Standard (DES) uses a Symmetric Encryption technique, in which the same key is applied at both sides (sender/receiver), known as the “Secret Key”
- It takes a fixed-length string of plaintext bits and transforms it through a series of complicated operations into another ciphertext bitstring of the same length.
- The secret key consists of 64 bits; however, only 56 of these are actually used by the encryption algorithm, since 8 bits are used solely for checking parity.
- DES was substituted only after 2001 with the advent of the AES (Advanced Encryption Standard) algorithm, which is based on the Rijndael developed by two Belgian cryptographers (Vincent Rijmen and Joan Daemen), who submitted a proposal to the US NIST during the AES selection process.
- Other famous symmetric encryption algorithms are: Triple-DES, CDMF (Commercial Data Masking Facility), IDEA (International Data Encryption Algorithm), RC2, RC4, RC5, RC6, MARS, Blowfish and Twofish.



Feistel also published a seminal article on Scientific American, in 1973.



It was the very first general-public information about Cryptography and Privacy.

There is growing concern that computers now constitute, or will soon constitute, a dangerous threat to individual privacy. Since many computers contain personal data and are accessible from distant terminals, they are

diverse jurisdictions. It will be argued here, however, that a computer system can be adapted to guard its contents from everyone but authorized individuals by enciphering the material in forms highly resistant to cipher-breaking.

There is no reason why the security system described for a single link could not be expanded to provide security for all users of a network. Other cryptographic approaches are still being studied in our laboratory and elsewhere. It would be surprising if cryptography, the traditional means of ensuring confidentiality in communication, could not provide privacy for a community of data-bank users.

After Feistel's work on LUCIFER and his article on Scientific American, many researchers became interested in the field of Cryptography.

and some of them decided to investigate on how to solve a fundamental problem:

**how to send the symmetric key to the
other side through a non-trusted
channel
?**



Martin Hellman

In late 1968 began work with AI at IBM's TJ Watson Research Center where he met members of the IBM cryptographic research program, including Horst Feistel and Alan G. Konheim, another IBM cryptographer.

"I remember many conversations with Feistel. That year at IBM and my interactions with Horst played a major role in my later moving into cryptography." Once he had lunch with Feistel, who talked not only about classical cryptographic systems as well as about the *"problems that sounded unsolvable but could actually be solved"*, as Feistel said.

So Hellman decided solve the Key Exchange Problem.

He left IBM and went to the MIT and then to the Electrical Engineering Faculty at Stanford University.



Whitfield Diffie

Mathematician from the Stanford University who started to study cryptography.

In the summer of 1974 Whitfield Diffie went to the IBM and spoke to Alan Konheim about challenges in the Cryptography field. Konheim was a little secretive but told him one very important thing:

“An old friend of mine, named Martin Hellman, worked here until some time ago, and now he's out at Stanford. And two people can work on a problem much better than one, and so when you get back to Stanford, you should look him up.”

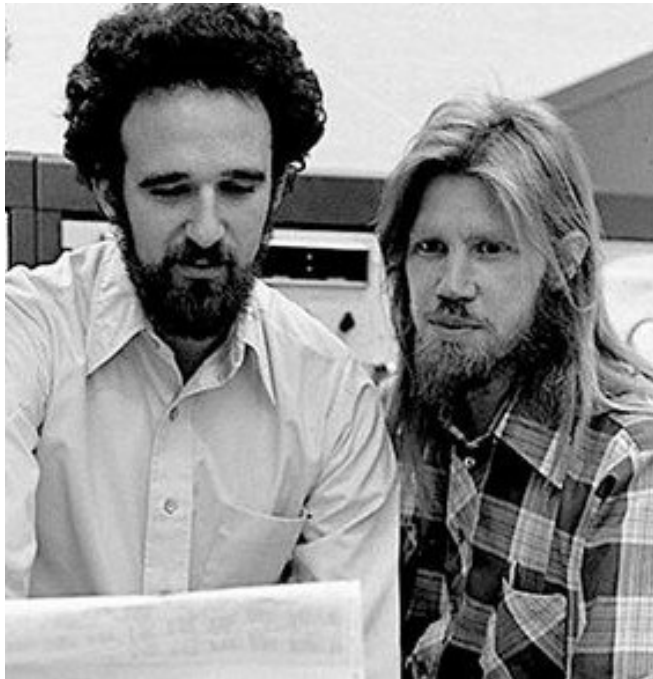


Alan Konheim, IBM



In late 1974, at the suggestion of Konheim, Diffie visited Hellman in Stanford. A planned half-hour early afternoon meeting between Diffie and Hellman evolved to an afternoon-long discussion followed by dinner and further exchange well into the evening. And it was just the beginning...

In Hellman's words, *“it was a mild epiphany, finding an intellectual soul mate.”*



Diffie and Hellman, 1976

The first public idea of an asymmetric public-private key cryptosystem

644

IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

"We stand today on the brink of a revolution in cryptography."

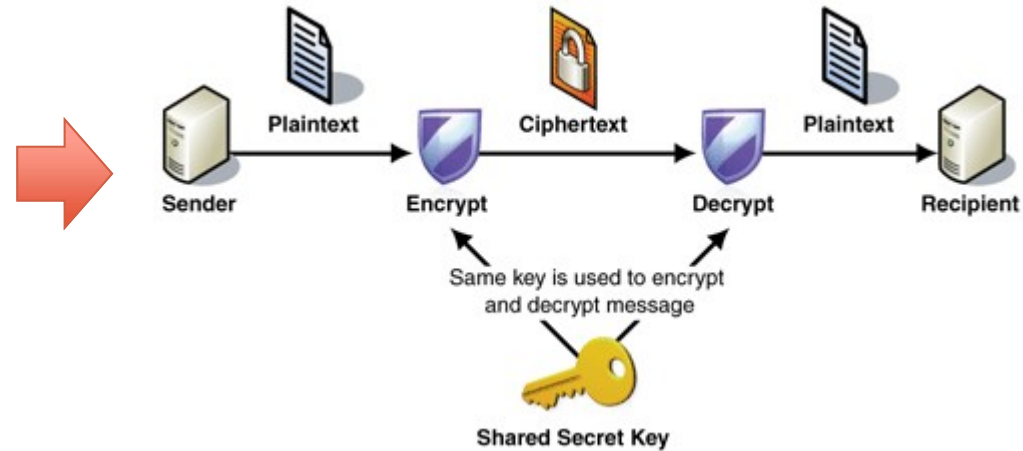
Abstract—Two kinds of contemporary developments in cryptography have given rise to a new approach to the problem of secure communication. The first is the development of techniques for the supply of the key to the receiver. The second is the development of techniques for the supply of the key to the sender. The theory of public key cryptography provides the theoretical basis for these techniques.

WE have given rise to a new approach to the problem of secure communication. The first is the development of techniques for the supply of the key to the receiver. The second is the development of techniques for the supply of the key to the sender. The theory of public key cryptography provides the theoretical basis for these techniques.

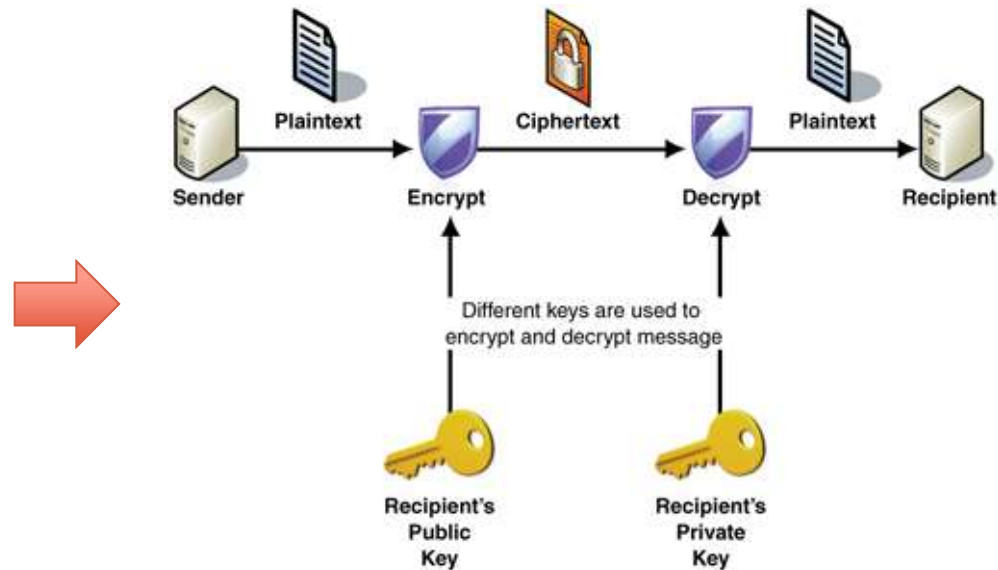
We propose some techniques for developing public key cryptosystems, but the problem is still largely open.

Public key distribution systems offer a different approach to eliminating the need for a secure key distribution channel. In such a system, two users who wish to exchange a key communicate back and forth until they arrive at a key in common. A third party eavesdropping on this exchange must find it computationally infeasible to compute the key from the information overheard. A possible solution to the public key distribution problem is given in Section III, and Merkle [1] has a partial solution of a different form.

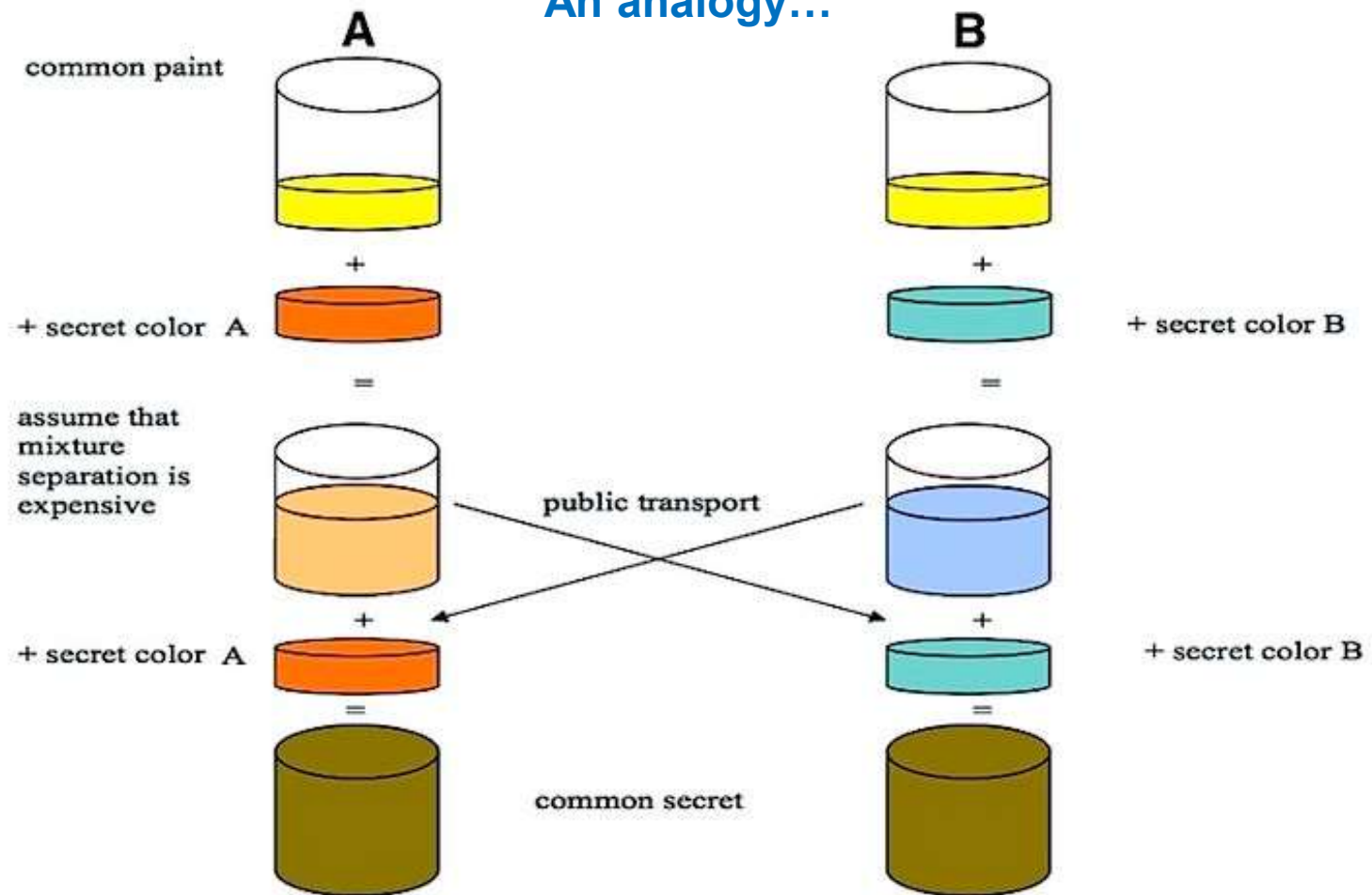
Symmetric Key Encryption (ex: DES Algorithm)



Asymmetric Key Encryption Ex: Diffie-Hellman



An analogy...



Public-key cryptography depends upon the existence of mathematical functions that are easy to compute whereas their inverse function is relatively difficult to do.

Example: *Exponentiation vs. logarithms*

Suppose you take the number 3 to the 6th power; again, it is relatively easy to calculate $3^6 = 729$. But if you start with the number 729 and need to determine the two integers, x and y so that $\log_x 729 = y$, it will take longer to find the two values.

WHITFIELD DIFFIE & MARTIN HELLMAN

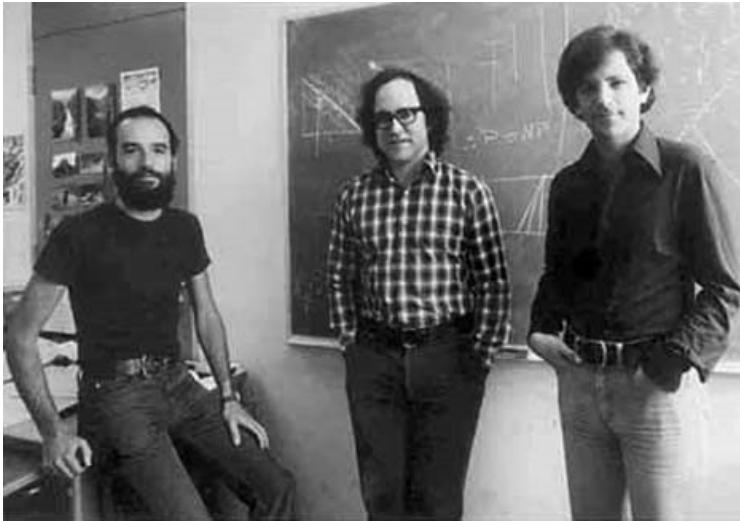
Invented public-key
cryptography



A.M.
TURING AWARD
2015



The dawn of the implementable Public-Key Cryptography, 1977



Ron Rivest, Adi Shamir, and Leonard Adleman from the Massachusetts Institute of Technology (MIT)

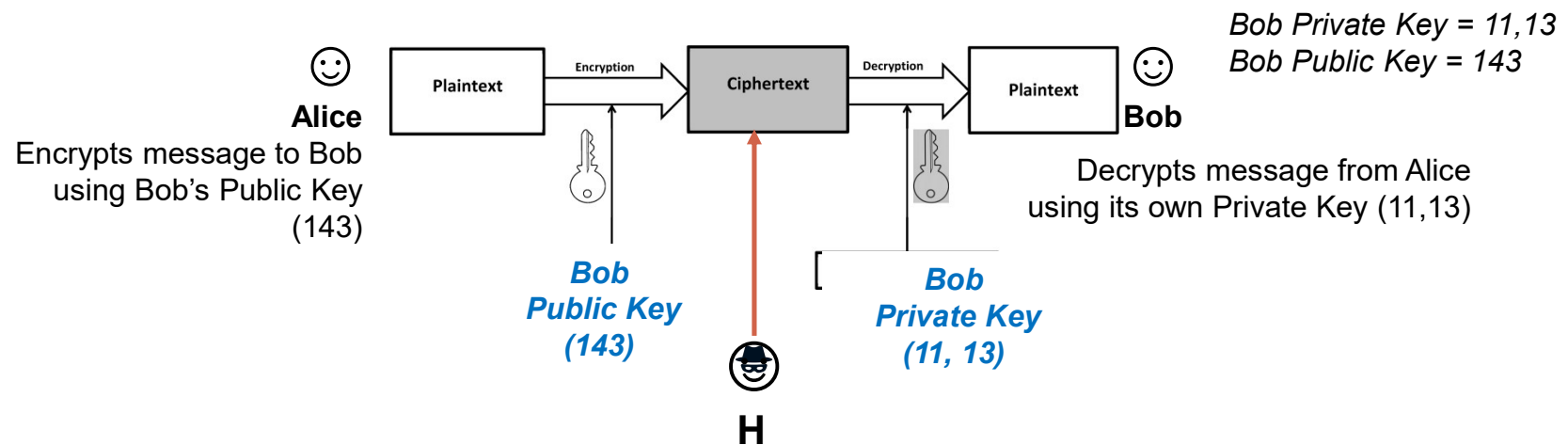
They invented (and patented) the **RSA** Algorithm which was the first public-key cryptosystems implemented and widely used for secure data transmission.

In RSA, this asymmetry is based on the practical difficulty of the factorization of the product of two large prime numbers, the "factoring problem".

- They filed the original RSA Patent with the US Patent Office on Dec 14, 1977, [U.S. Patent 4,405,829](#).
- They founded RSA Data Security in 1982.
- In 1996 Security Dynamics acquired RSA Data Security but kept the name (RSA Security).
- In 1997 RSA Security proposed the first of the DES Challenges which led to the first public breaking of DES.
- In 2006 EMC acquired RSA Security.
- In 2016 Dell merged with EMC and RSA Security became a subsidiary of Dell Technologies
- On Sept 2020 Symphony Technology Group (STG) acquired RSA Security from Dell Technologies

RSA explanation (tried to make it easy)

Suppose you have two prime numbers, 11 and 13, and you need to calculate the product; it should take almost no time to calculate that value, which is 143. Now suppose, instead, that you have a number that is a product of two primes, 143, and you need to determine those prime factors. You will eventually come up with the solution but whereas calculating the product took milliseconds, factoring will take much longer. That's it.



Hacker will know the message was encrypted with B's Public Key (143) and will have to factor this number to find the two prime numbers that were multiplied to generate it

Reality is about random BIG prime numbers (example)

Private
Key

```
3347807169895689878604416984821269081770479498371376856891
2431388982883793878002287614711652531743087737814467999489
★
3674604366679959042824463379962795263227915816434308764267
6032283815739666511279233373417143396810270092798736308917
```

=

Public
Key

```
1230186684530117755130494958384962720772853569595334792197
3224521517264005072636575187452021997864693899564749427740
6384592519255732630345373154826850791702612214291346167042
9214311602221240479274737794080665351419597459856902143413
```

This RSA-768 public key above, which has 232 decimal digits (768 bits), was factored in a test in 2009 over the span of two years, by a group of 15 people using a collection of parallel computers amounted approximately to the equivalent of almost 2000 years of computing time on a single-core 2.2 GHz AMD Opteron-based computer.

RON RIVEST, ADI SHAMIR & LEN ADLEMAN



RSA public-key cryptography

A.M.
TURING

The (secret) work done in the UK

For some time, it was a quiet secret that a team at the UK's Government Communications Headquarters (GCHQ) had first developed Public Key Cryptography in the early 1970s.

Because of the nature of the work, GCHQ kept the original memos classified. In the 90s, however, the GCHQ changed their posture when they realized that there was nothing to gain by continued silence.

Documents show that a GCHQ mathematician named James Ellis started research into the key distribution problem in 1969 and that by 1975, James Ellis, Clifford Cocks, and Malcolm Williamson had worked out all of the fundamental details of PKC, yet couldn't talk about their work. By 1999, Ellis, Cocks, and Williamson began to get their due credit in a break-through article in *WIRED Magazine* (<https://www.wired.com/1999/04/crypto/>)



STEVEN LEVY 04.01.99 12:00 PM

THE OPEN SECRET

Public key cryptography - the breakthrough that revolutionized email and ecommerce - was first discovered by American geeks. Right? Wrong.

So roll back the time frame to a few years before the Diffie-Hellman/RSA saga. Change the setting from MIT and Stanford, places where you can nearly get sunburned from the heat of new ideas, to a cloistered British intelligence agency in the sleepy Cotswolds city of Cheltenham, about 100 miles from London.



James Ellis

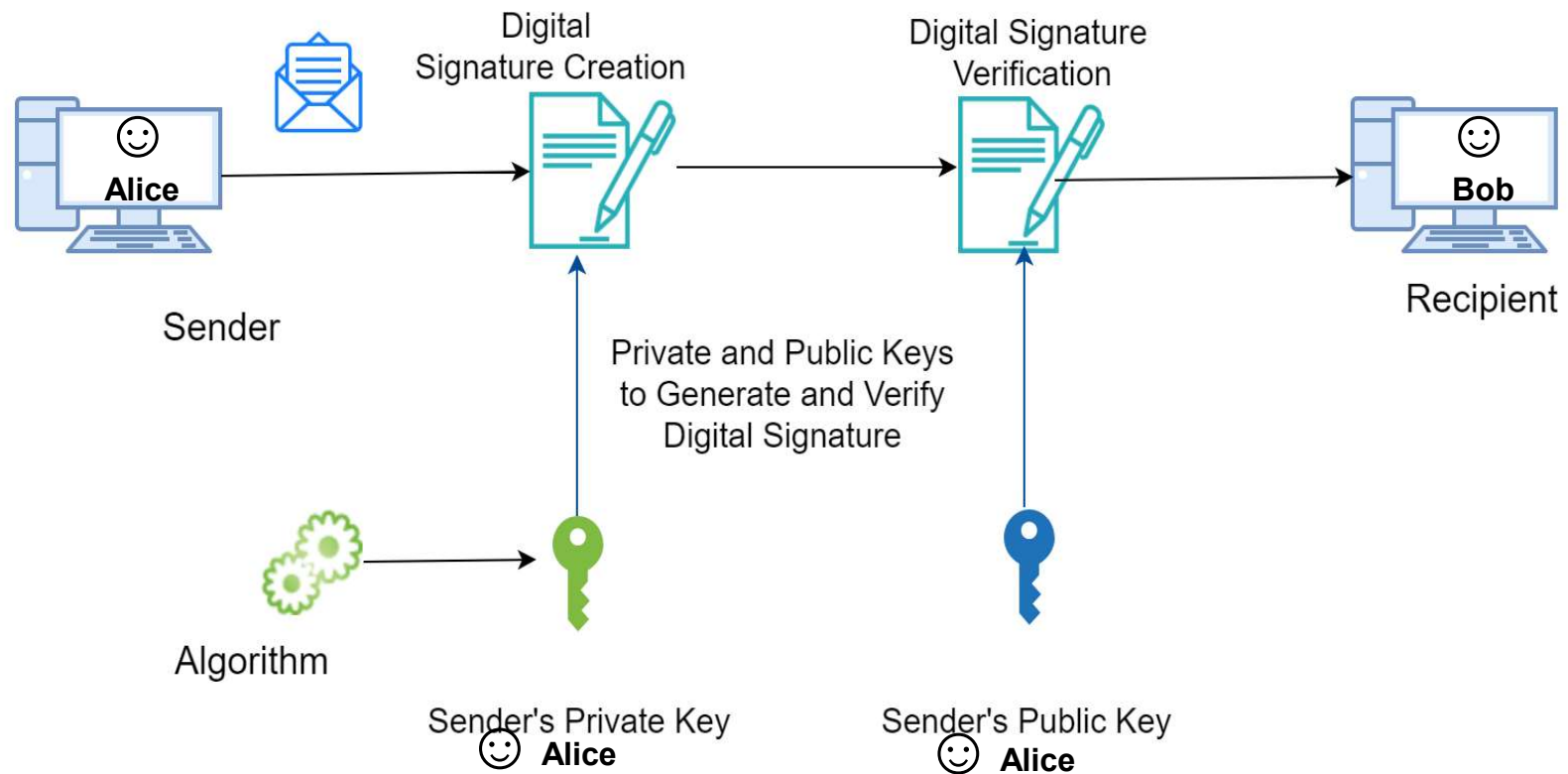


Clifford Cocks



Malcolm Williamson

The other way around of using Public/Private Keys: Digital Signature



Primality test using Elliptic Curve, 1985

A **primality test** is an algorithm for determining whether an input number is prime, which is important to key generation, that's why among other fields of mathematics, it is used for cryptography.

Unlike integer factorization, primality tests do not generally give prime factors, only stating whether the input number is prime or not.

H. W. Lenstra developed the concept of using elliptic curves in factorization in 1985, and the implications for its use in primality testing (and proving) followed quickly.



Hendrik Lenstra
University of Amsterdam

Elliptic-Curve Cryptography (ECC), 1985

Inspired by the work of Lenstra, Elliptic-Curve Cryptography (ECC) was developed as an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite numeric fields.

It proposed independently by Neal Koblitz (UW) and Victor Miller (IBM), in 1985.

Elliptic curve cryptography algorithms entered wide use after 2005. It requires smaller keys compared to RSA to provide equivalent security.

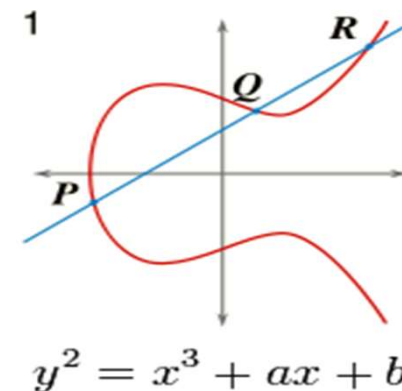
RSA Key Size (bits)	ECC Key Size (bits)
1024	160
2048	224
3072	256
7680	384
15360	512



Neal Koblitz
Univ. Washington



Victor Miller
IBM

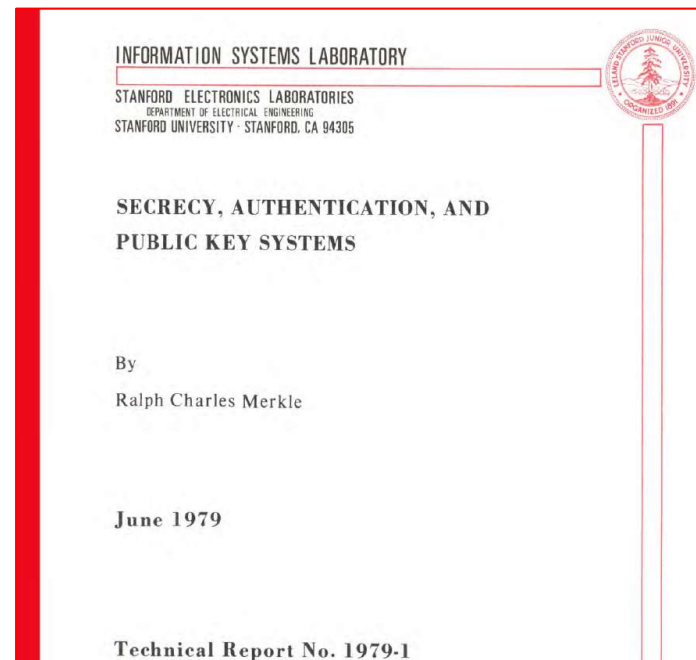
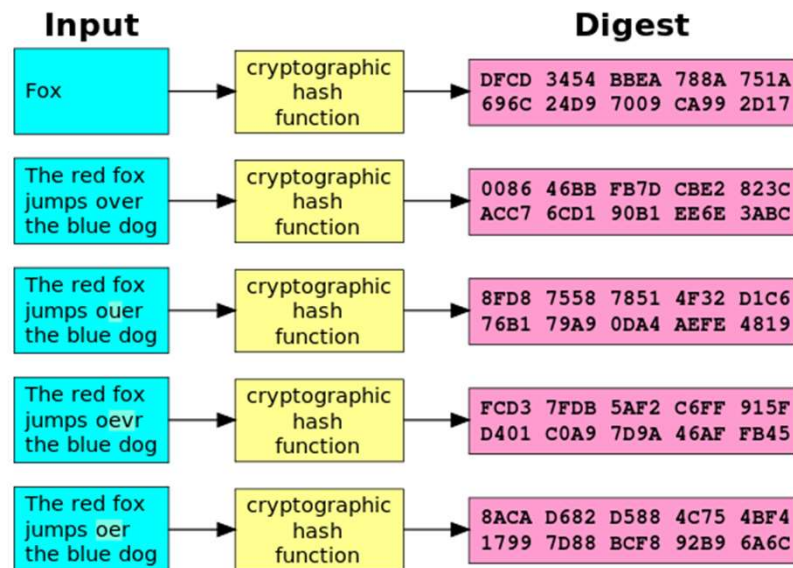


Cryptographic Hash 1979

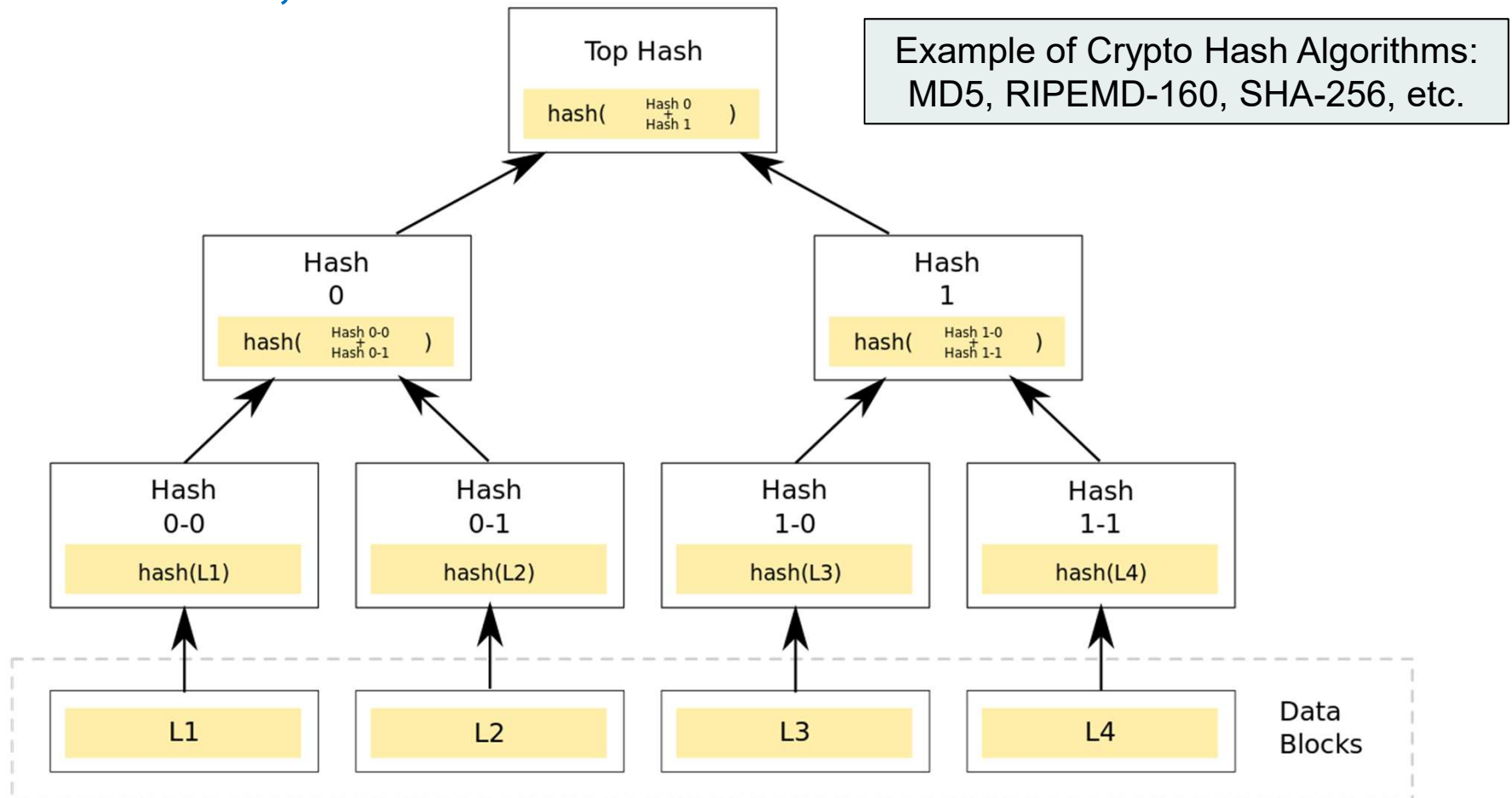
Ralph Merkle
Stanford University



The cryptographic hash mathematical algorithm that maps data of arbitrary size to a bit string of a fixed size (a hash) and is designed to be a one-way function, that is, a function which is infeasible to invert.



Merkle Tree, 1979



Computational Proof of Work (1990s)

In 1992 email spam messaging was already a problem. Two researchers **Cynthia Dwork** (IBM) and **Moni Naor** (Weizmann IS), tried to solve it. The essay, “*Pricing via Processing or Combatting Junk Mail*,” presented a way to prevent spammers from sending out unsolicited mass messages.



Adam Back

In 1997, a member of the cypherpunk movement named **Adam Back** founded a protocol called **HashCash**. Many of the ideas presented with the HashCash protocol evolved into what we understand to be a Proof of Work mechanism today, including something called “Double Spending Protection,” a foundational concept in blockchain.

In 1999 **Markus Jakobsson** (USCD) and **Ari Juels** (RSA) wrote a paper called “Proof of Work and Bread Pudding Protocols,” thus coining the term, defined as a protocol in which a prover demonstrates to a verifier that has expended a certain level of computational effort in a specified interval of time.

Pricing via Processing OR Combatting Junk Mail*

Cynthia Dwork * Moni Naor †

Abstract

We present a computational technique for combatting junk mail, in particular, and controlling access to a shared resource, in general. The main idea is to require a user to compute a moderately hard, but not intractable, function in order to gain access to the resource, thus preventing frivolous use. To this end we suggest several *pricing functions*, based on, respectively, extracting square roots modulo a prime, the Fiat-Shamir signature scheme, and the Ong-Schnorr-Shamir (cracked) signature scheme.



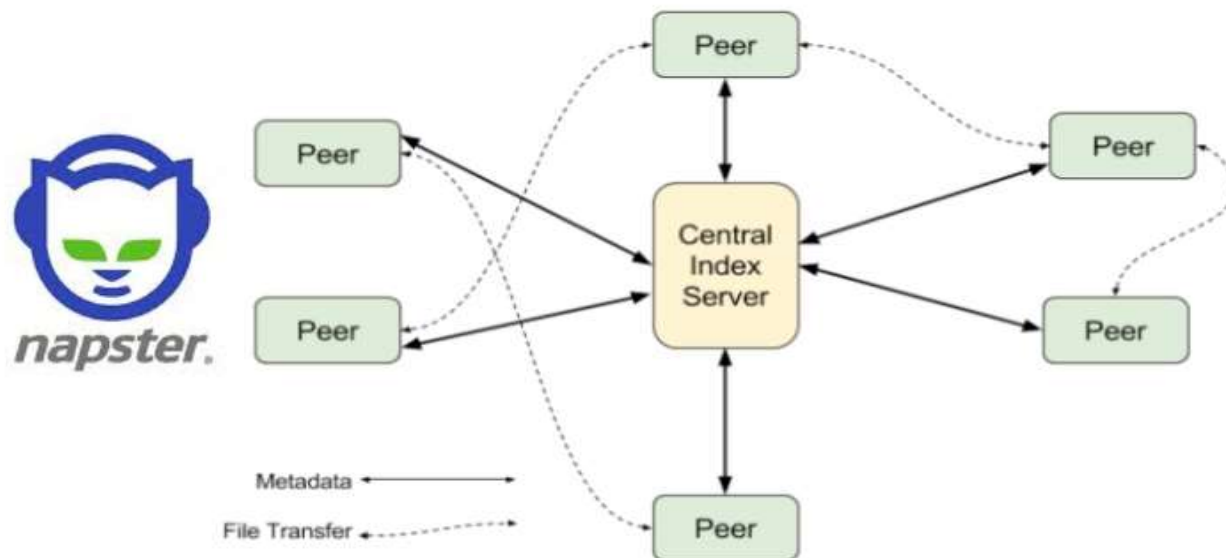
Markus Jakobsson



Ari Juels

Peer-to-Peer Networks (1999 and 2000s)

Peer-to-Peer (P2P) computing or networking - which has nothing to do with Cryptography - is a distributed application architecture that partitions tasks or workloads between peers, which are equally privileged, equipotent participants in the application network. Peers make a portion of their resources, such as processing power, disk storage or network bandwidth, directly available to other network participants, without the need for central coordination by servers or stable hosts. Peers are both suppliers and consumers of resources, in contrast to the traditional client-server model in which the consumption and supply of resources is divided. This idea started with Napster in 1999, which had a central indexing server to index the contents of all of the peers that register with it. Napster was closed in 2001 but many other P2P networks appeared on the Internet.



The birth of Bitcoin, 2008

- The Internet domain name "bitcoin.org" was registered on 18 August 2008.
- On 31 October 2008, a link to a paper authored by Satoshi Nakamoto titled *Bitcoin: A Peer-to-Peer Electronic Cash System* was posted to a cryptography mailing list.
- Nakamoto implemented the bitcoin software as open-source code and released it in January 2009.
- Nakamoto's identity remains unknown.



Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

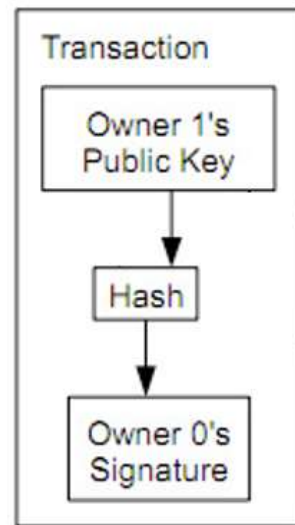
Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

The paper detailed methods of using a peer-to-peer network to generate what was described as "a system for electronic transactions without relying on trust".

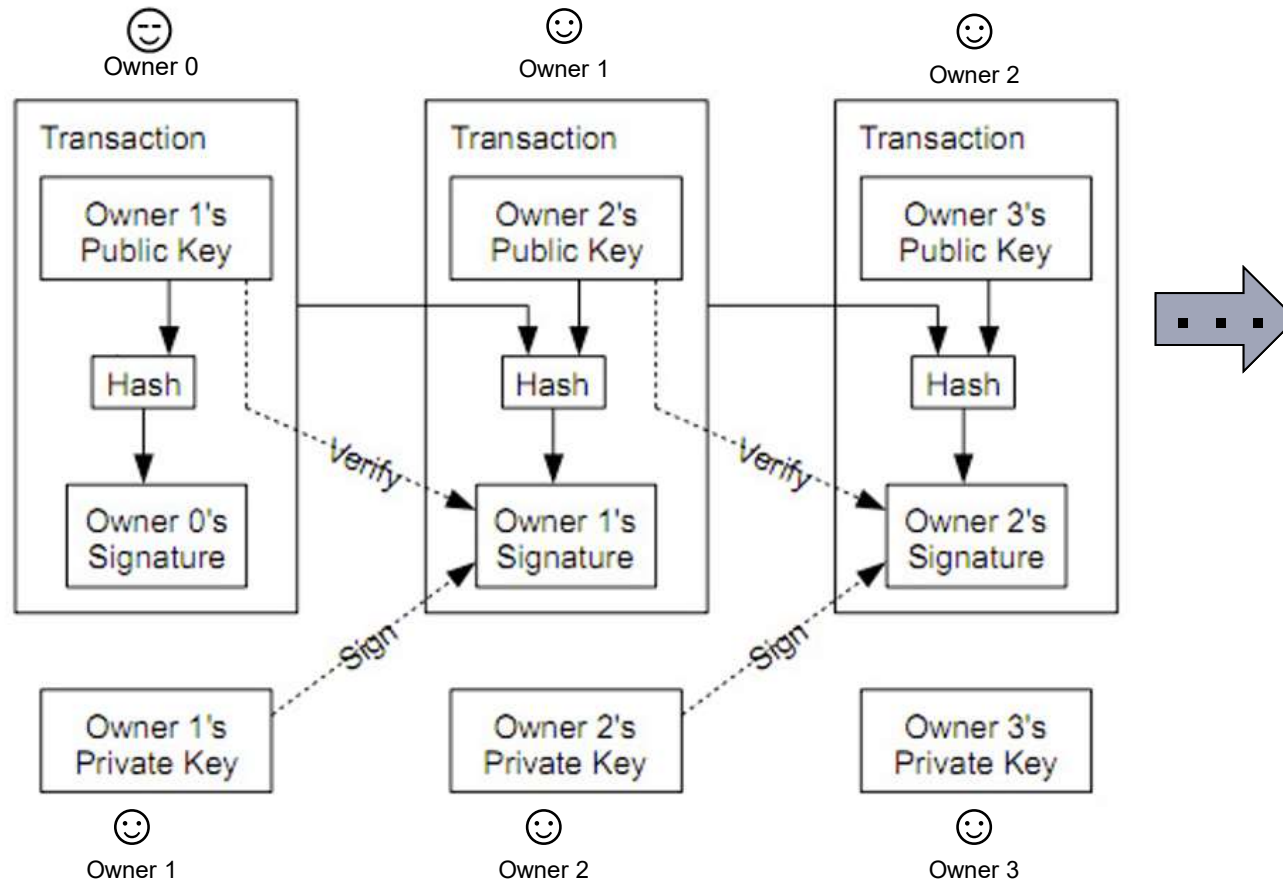
Satoshi created great innovation combining quite old things:

Cryptography (Public-Private keys/Elliptic Curve Cryptography, Digital Signatures, Hash & Merkle Trees), from the 70s/80s, Proof of Work (90s) and P2P Networks from 90s and 2000s

A Bitcoin transaction

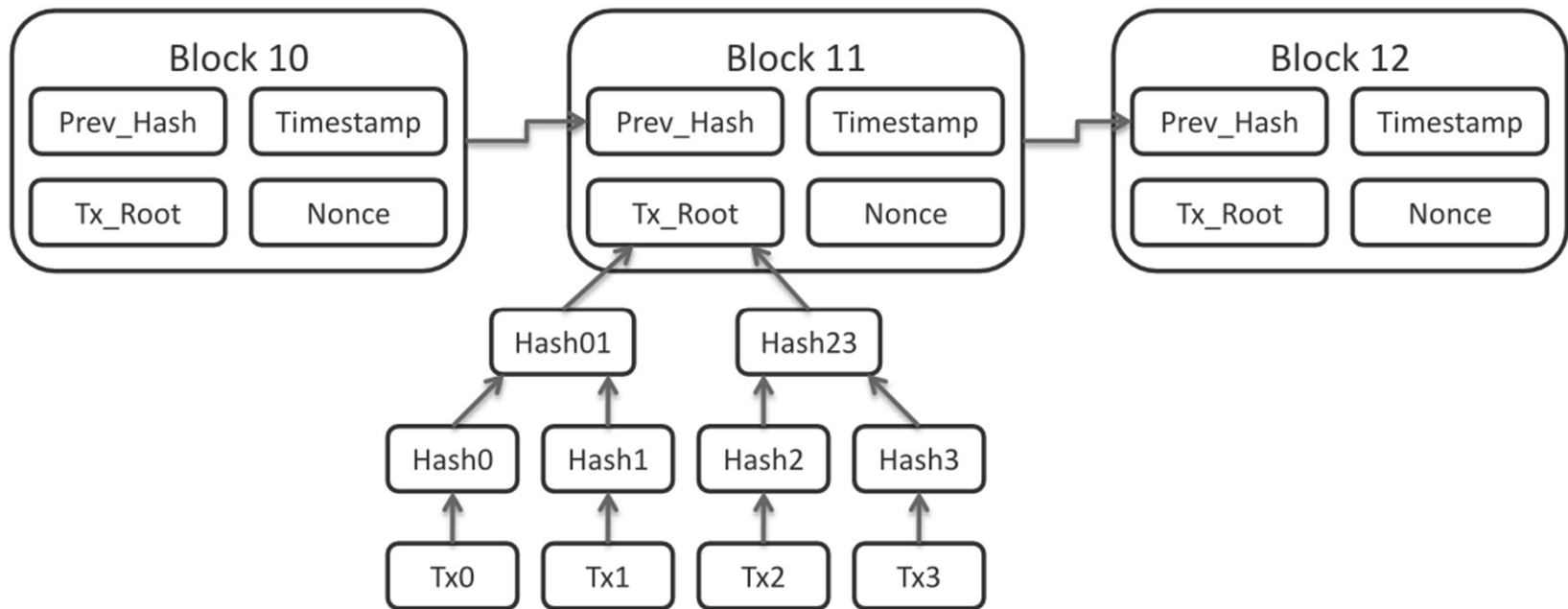


A Chain of Transactions Blocks → Blockchain



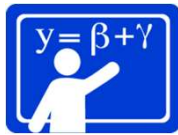


Blocks



Transactions

Some examples of consensus algorithms



Proof of work



Proof of stake



Solo /
No-ops



Kafka /
Zookeeper



Proof of
Elapsed Time



PBFT
(Byzantine fault
tolerance)based

Consensus algorithms have different strengths and weaknesses



Proof of work

Require validators to solve difficult cryptographic puzzles

PROs: Works in untrusted networks

CONS: Relies on energy use; slow to confirm transactions

Example usage: Bitcoin, Ethereum



Proof of stake

Require validators to hold currency in escrow

PROs: Works in untrusted networks

CONS: Requires intrinsic (crypto)currency, "Nothing at stake" problem

Example usage: Nxt



Proof of
Elapsed Time

Wait time in a trusted execution environment randomizes block generation

PROs: Efficient

CONS: Requires processor extensions

Example usage: Hyperledger Sawtooth

Consensus algorithms have different strengths and weaknesses



Solo /
No-ops

Validators apply received transactions without consensus

PROs: Very quick; suited to development

CONS: No consensus; can lead to divergent chains

Example usage: Hyperledger Fabric V1



PBFT-based

Practical Byzantine Fault Tolerance implementations

PROs: Reasonably efficient and tolerant against malicious peers

CONS: Validators are known and totally connected

Example usage: Hyperledger Fabric V0.6



Kafka /
Zookeeper

Ordering service distributes blocks to peers

PROs: Efficient and fault tolerant

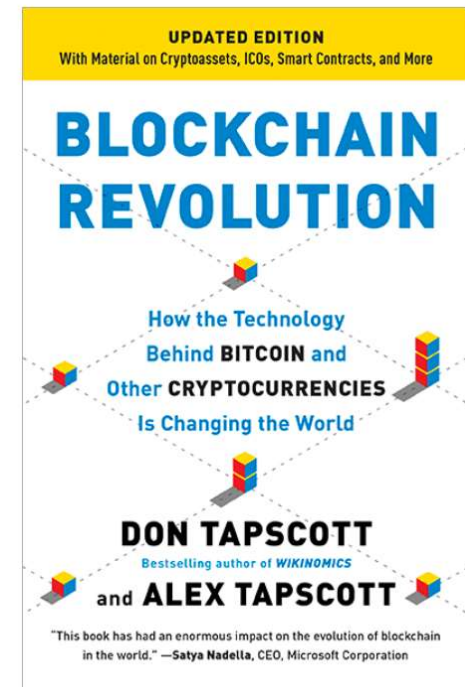
CONS: Does not guard against malicious activity

Example usage: Hyperledger Fabric V1

A Blockchain for Business Reference (Don Tapscott)



<https://www.blockchainresearchinstitute.org/>



<http://blockchain-revolution.com/>

A Technical Reference for Blockchain

Dr. C. Mohan



Distinguished Visiting Professor, Tsinghua University, Beijing

Advisor, Kerala Blockchain Academy & Tamil Nadu e-Governance Agency, India

Retired IBM Fellow (1997-2020), IBM Research (1981-2020), San Jose, USA.

- Links to Videos, Slides, etc. <http://bit.ly/CMbcDB> →
- Telegram, Twitter, Instagram: [@seemohan](#)
- Facebook: <http://www.facebook.com/cmohan>
- LinkedIn: <http://www.linkedin.com/in/seemohan/>

Videos and Slides of My Conference Keynotes, Tutorials and other Talks

Slides (latest) of my presentation “New Era in Distributed Computing with Blockchains and Databases” at the Hot Chips Conference on 19 August 2018 in Cupertino are at <http://bit.ly/HotcBC> Announcement: <http://bit.ly/HotCbc>

Slides (latest) of my presentation “Blockchain: Introduction and Career Options” at the BITS Alumni Association Silicon Valley Chapter Tech Day and Career Fair on 30 June 2018 in Milpitas are at <http://bit.ly/bitsBC> Announcement: <http://bit.ly/BITSbc>

Slides (latest) of my presentation “Introduction to Hyperledger Fabric and IBM Blockchain Platform” at the World Blockchain Hackathon in San Francisco (USA) on 16 June 2018 are at <http://bit.ly/wbhCMP> Announcement: <http://bit.ly/CMbcDB>

Slides (latest) and **Video (latest)** of my presentation “Blockchains Untangled: Public, Private, Smart Contracts, Applications, Issues” at the University of California in Irvine (USA) on 1 June 2018 are at <http://bit.ly/pBCuci> and <http://bit.ly/vbcUCI>, respectively. Talk Announcement: <http://bit.ly/UCIbca>

Slides of my presentation “Landscape of Practical Blockchain Systems and their Applications” at the 23rd International Conference on Database Systems for Advanced Applications (DASFAA) in Gold Coast (Australia) on 23 May 2018 are at <http://bit.ly/dAsFap>

Video and Slides of my Blockchain Track Keynote presentation “Blockchain Platform and Disruption Trends” at TiE Inflect 2018 in Santa Clara (USA) on 4 May 2018 are at <http://bit.ly/BcTiVi> and <http://bit.ly/TieBTp> Talk Announcement: <http://bit.ly/tieIBC>

Video and Slides of my Blockchain Bootcamp presentation “Landscape of Practical Blockchain Systems and their Applications” at TiE Inflect 2018 in Santa Clara (USA) on 3 May 2018 are at <http://bit.ly/TiBoVi> and <http://bit.ly/BcTiEi> Talk Announcement: <http://bit.ly/bootBC>

Slides of my presentation “New Era in Distributed Computing with Blockchains and Databases” at San Ramon Blockchain Meetup in San Ramon (USA) on 3 March 2018 are at <http://bit.ly/BRramo> Talk Abstract: <http://bit.ly/SanRBC>



**...but what about quantum
computers breaking the existing
cryptographic systems?**

The three known types of quantum computing and their **applications, generality, and computational power.**



A very specialized form of quantum computing with unproven advantages over other specialized forms of conventional computing.

DIFFICULTY LEVEL



The most likely form of quantum computing that will first show true quantum speedup over conventional computing. This could happen within the next five years.

DIFFICULTY LEVEL



The true grand challenge in quantum computing. It offers the potential to be exponentially faster than traditional computers for a number of important applications for science and businesses.

DIFFICULTY LEVEL



Quantum Annealer

The quantum annealer is least powerful and most restrictive form of quantum computers. It is the easiest to build, yet can only perform one specific function. The consensus of the scientific community is that a quantum annealer has no known advantages over conventional computing.

APPLICATION
Optimization Problems

GENERILITY
Restrictive

COMPUTATIONAL POWER
Same as traditional computers

Analog Quantum

The analog quantum computer will be able to simulate complex quantum interactions that are intractable for any known conventional machine, or combinations of these machines. It is conjectured that the analog quantum computer will contain somewhere between 50 to 100 qubits.

APPLICATIONS
Quantum Chemistry
Material Science
Optimization Problems
Sampling
Quantum Dynamics

GENERILITY
Partial

COMPUTATIONAL POWER
High

Universal Quantum

The universal quantum computer is the most powerful, the most general, and the hardest to build, posing a number of difficult technical challenges. Current estimates indicate that this machine will comprise more than 100,000 physical qubits.

APPLICATIONS
Secure computing
Machine Learning
Cryptography
Quantum Chemistry
Material Science
Optimization Problems
Sampling
Quantum Dynamics
Searching

GENERILITY
Complete with known speed up

COMPUTATIONAL POWER
Very High

The Shor Algorithm (1994)



Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor[†]

Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

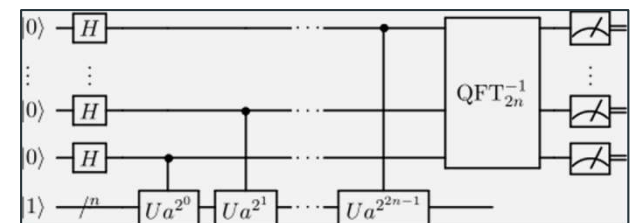
Keywords: algorithmic number theory, prime factorization, discrete logarithms, Church's thesis, quantum computers, foundations of quantum mechanics, spin systems, Fourier transforms

AMS subject classifications: 81P10, 11Y05, 68Q10, 03D10

Created by the mathematician Peter Shor (now MIT) while he was working for AT&T Research Labs.


It is an algorithm that runs on a quantum computer for solving the integer factorization problem:

given an integer N , find its prime factors.



Yes, RSA can be broken with Quantum Computers

The Shor Algorithm for ECC (2003)

 Cornell University
Library

arXiv.org > quant-ph > arXiv:quant-ph/0301141

Quantum Physics

Shor's discrete logarithm quantum algorithm for elliptic curves

John Proos, Christof Zalka

(Submitted on 25 Jan 2003 (v1), last revised 22 Jan 2004 (this version, v2))

We show in some detail how to implement Shor's efficient quantum algorithm for discrete logarithms for the particular case of elliptic curve groups. It turns out that for this problem a smaller quantum computer can solve problems further beyond current computing than for integer factorisation. A 160 bit elliptic curve cryptographic key could be broken on a quantum computer using around 1000 qubits while factoring the security-wise equivalent 1024 bit RSA modulus would require about 2000 qubits. In this paper we only consider elliptic curves over $GF(p)$ and not yet the equally important ones over $GF(2^n)$ or other finite fields. The main technical difficulty is to implement Euclid's gcd algorithm to compute multiplicative inverses modulo p . As the runtime of Euclid's algorithm depends on the input, one difficulty encountered is the "quantum halting problem".

Comments: 34 pages Latex, essentially published version, but not using Journal style file

Subjects: **Quantum Physics (quant-ph)**

Journal reference: QIC 3 (No. 4) (2003) pp.317-344

Cite as: **arXiv:quant-ph/0301141**

Yes, ECC can be broken with Quantum Computers

But there is also Quantum Cryptography

Quantum key distribution (QKD) is a secure communication method which implements a cryptographic protocol involving components of quantum mechanics. It enables two parties to produce a shared random secret key known only to them, which can then be used to encrypt and decrypt messages. It is often called **quantum cryptography**.

BB84 was the first QKD scheme developed by Charles Bennett (IBM) and Gilles Brassard (Université de Montréal) in **1984**. There are others schemas.



Charles Bennett
IBM



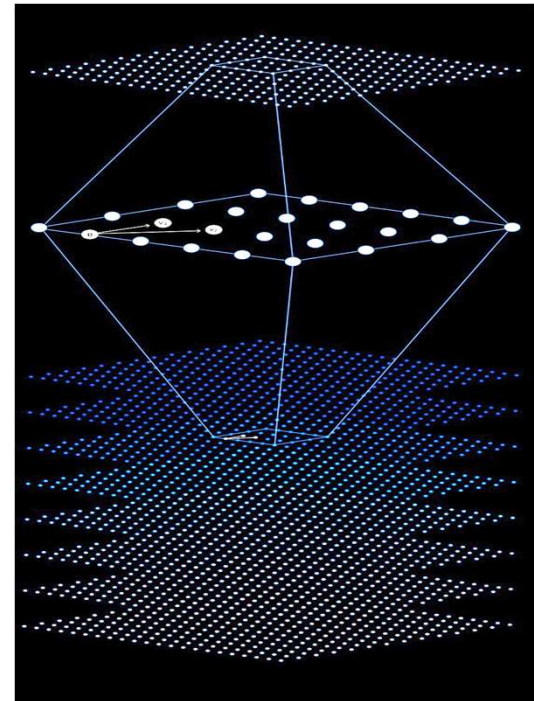
Gilles Brassard
Université de Montréal

Alice's random bit	0	1	1	0	1	0	0	1
Alice's random sending basis	+	+	×	+	×	×	×	+
Photon polarization Alice sends	↑	→	↘	↑	↘	↗	↗	→
Bob's random measuring basis	+	×	×	×	+	×	+	+
Photon polarization Bob measures	↑	↗	↘	↗	→	↗	→	→
PUBLIC DISCUSSION OF BASIS								
Shared secret key	0		1			0		1

The Future?

Lattice Cryptography and Fully Homomorphic Encryption (FHE).

- IBM researchers are developing a new security method built on an underlying architecture known as lattice cryptography, which hides data inside complex math problems (algebraic structures) called lattices. The difficulty in solving these math problems is useful for cryptographers, because they can apply this intractability to protect information.
- Unlike commonly-used cryptosystems like RSA and ECC, lattice-based cryptosystems cannot feasibly (as far as we know) be broken by quantum computers. The NSA cited this as a reason to begin transitioning to lattice-based cryptosystems, or to other "post-quantum" cryptosystems.
- Lattice-based cryptography the basis of another encryption technology called Fully Homomorphic Encryption (FHE) – see next slide.



<https://www.research.ibm.com/5-in-5/lattice-cryptography/>

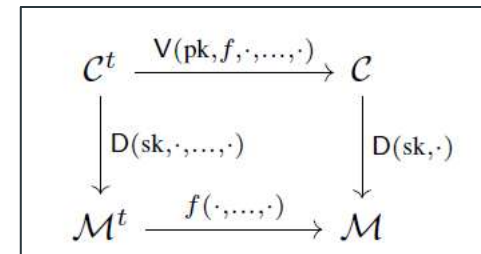
The Future?

Lattice Cryptography and Fully Homomorphic Encryption (FHE).

- **Homomorphic encryption** is a form of encryption that allows computation on ciphertexts, generating an encrypted result which, when decrypted, matches the result of the operations as if they had been performed on the plaintext.
- In 2009, Craig Gentry constructed the first Fully Homomorphic Encryption (FHE) scheme, which allows data to be processed in arbitrarily complex ways while it remains encrypted, solving a major open problem that had been unsolved for 30 years.
- FHE allows data processing to be outsourced (e.g., to the cloud) without sacrificing privacy - in particular, without disclosing the decryption key.



Craig Gentry
IBM



Why is this a problem today ?

Security time value of data

Certain data stored today
must remain confidential for
decades

HIPAA – 6 years from its last use, Securities exchange act
Tax Records 7-10 Years in most countries, Sarbanes Oxley
Guide 0068 - Clinical Trials 25 Years
Toxic Substances Control Act / Occupational Safety and Health A
Trade secrets , Mergers and Acquisitions,
Medical Records in Japan



Infrastructure Update Cycles

Many applications and
infrastructures have long
update cycles

Passports – 10 years from issue
Road Vehicles 15-20 years
Critical Infrastructures 25-30 Years
Aircraft/Trains 25-30 Years
Some critical mainframe applications (50 Years)





Thank You!

Marcelo Sávio

<https://www.linkedin.com/in/msavio>