POST-QUANTUM VERKLE SIGNATURE SCHEME

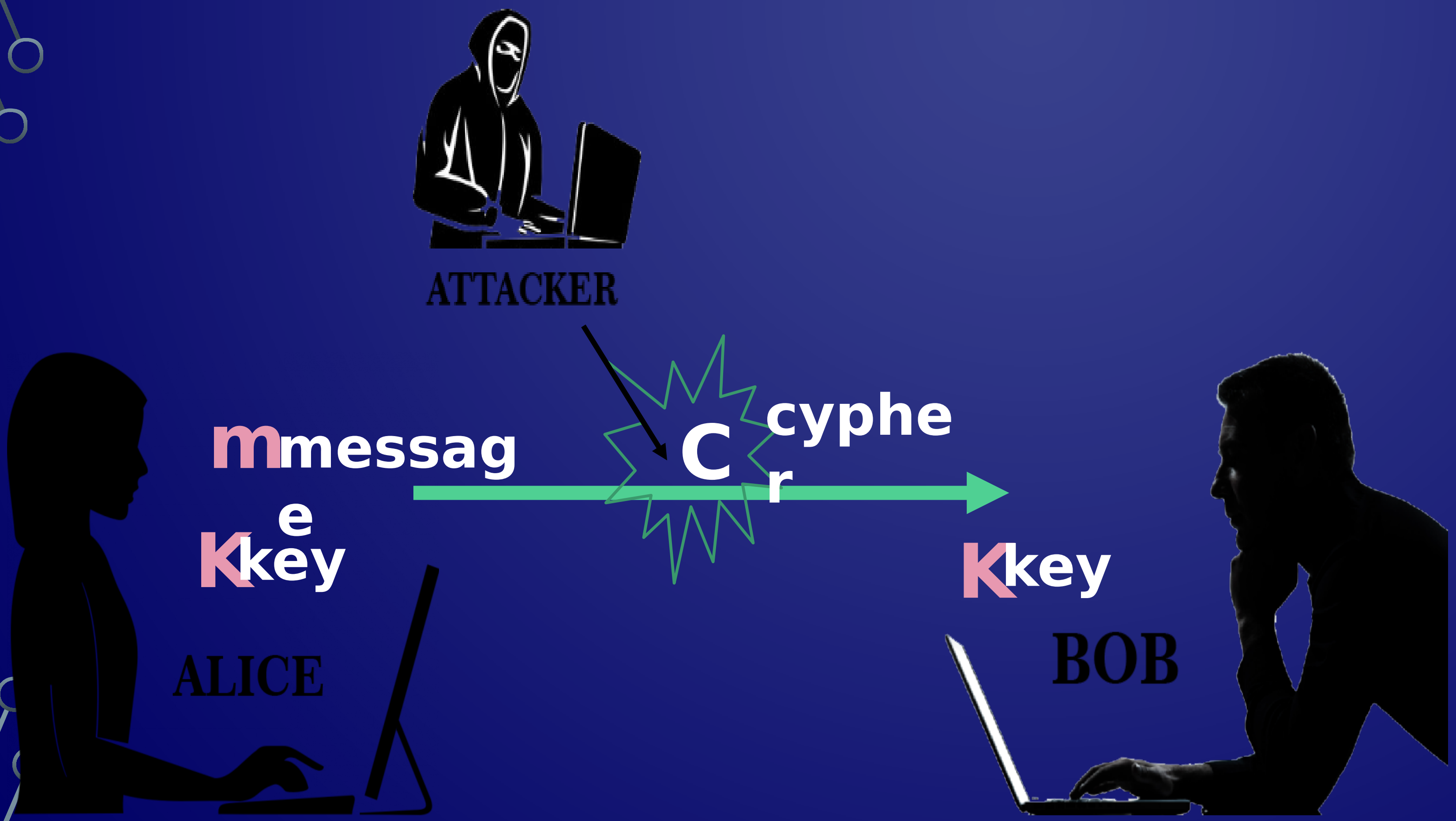
SCIENTIFIC CYBER SECURITY ASSOCIATION

კავკასიის უნივერსიტეტი
CAUCASUS UNIVERSITY

DEEPSEC

MAKSIM IAVICH

PRIVATE-KEY CRYPTOGRAPHY



$c := \mathrm{Enc}_k(m)$

Encryption

$\mathrm{Dec}_k(\mathrm{Enc}_k(m)) = m$

$m := \mathrm{Dec}_k(c)$

Decryption

PRIVATE-KEY CRYPTOGRAPHY

- Private-key cryptography allows two users who *share a secret key* to establish a “secure channel”
- The need to share a secret key incurs several drawbacks…

THE KEY-DISTRIBUTION PROBLEM

- *How do users share a key in the first place?*
  - Need to share the key using a secure channel…

- This problem can be solved in some settings…
  - E.g., physical proximity, trusted courier
  - (Note: this does not make private-key cryptography useless)
- …but not others (or at least not cheaply)

THE KEY-MANAGEMENT PROBLEM

- Imagine an organization with N employees, where each pair of employees might need to communicate securely

- Solution using private-key cryptography:
  - Each user shares a key with all other users

  $\Rightarrow$ Each user must store/manage N-1 secret keys!

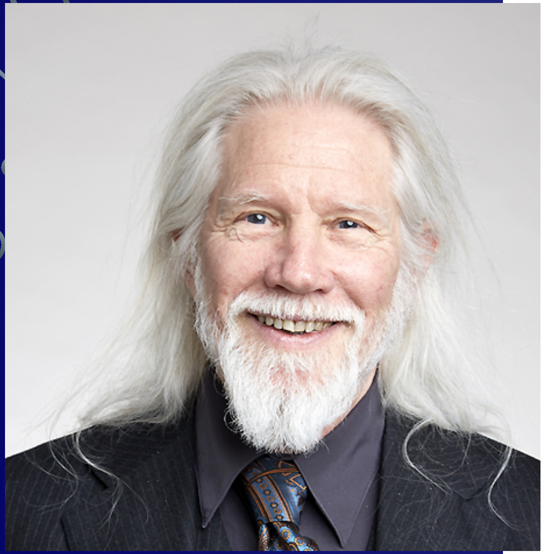
  $\Rightarrow$ $O(N^2)$ keys overall!

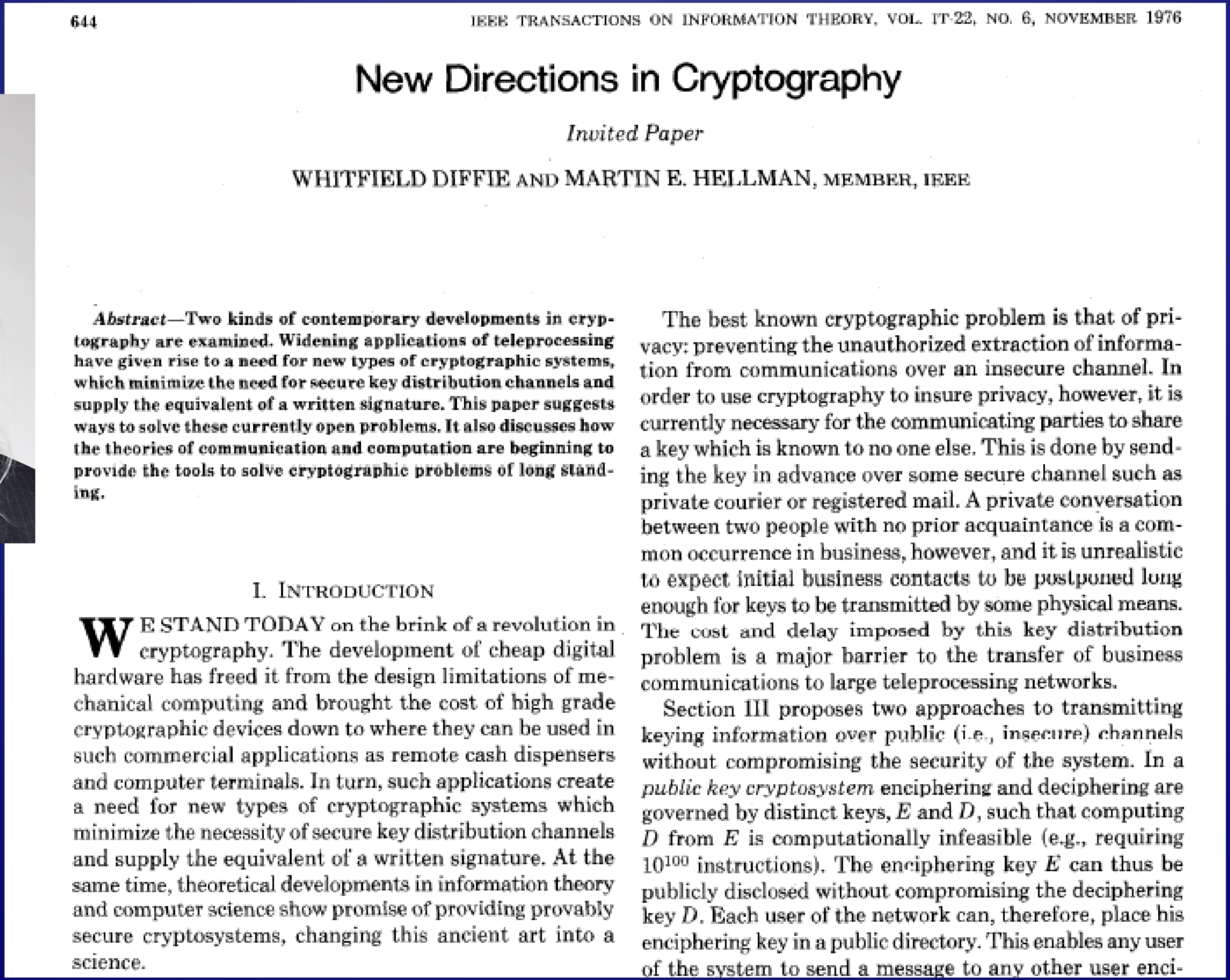
LACK OF SUPPORT FOR “OPEN SYSTEMS”

- Say two users *who have no prior relationship* want to communicate securely
  - When would they ever have shared a key?

- This is not at all far-fetched!
  - Customer sending credit-card data to merchant
  - Sending an email to a colleague

“CLASSICAL” CRYPTOGRAPHY OFFERS NO SOLUTION TO THESE PROBLEMS!





New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

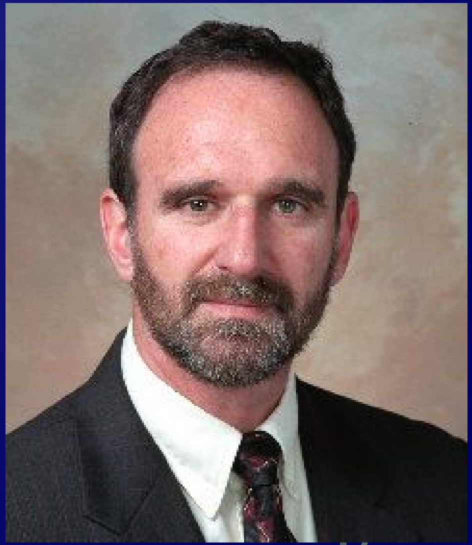
***Abstract*—Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.**

I. Introduction

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

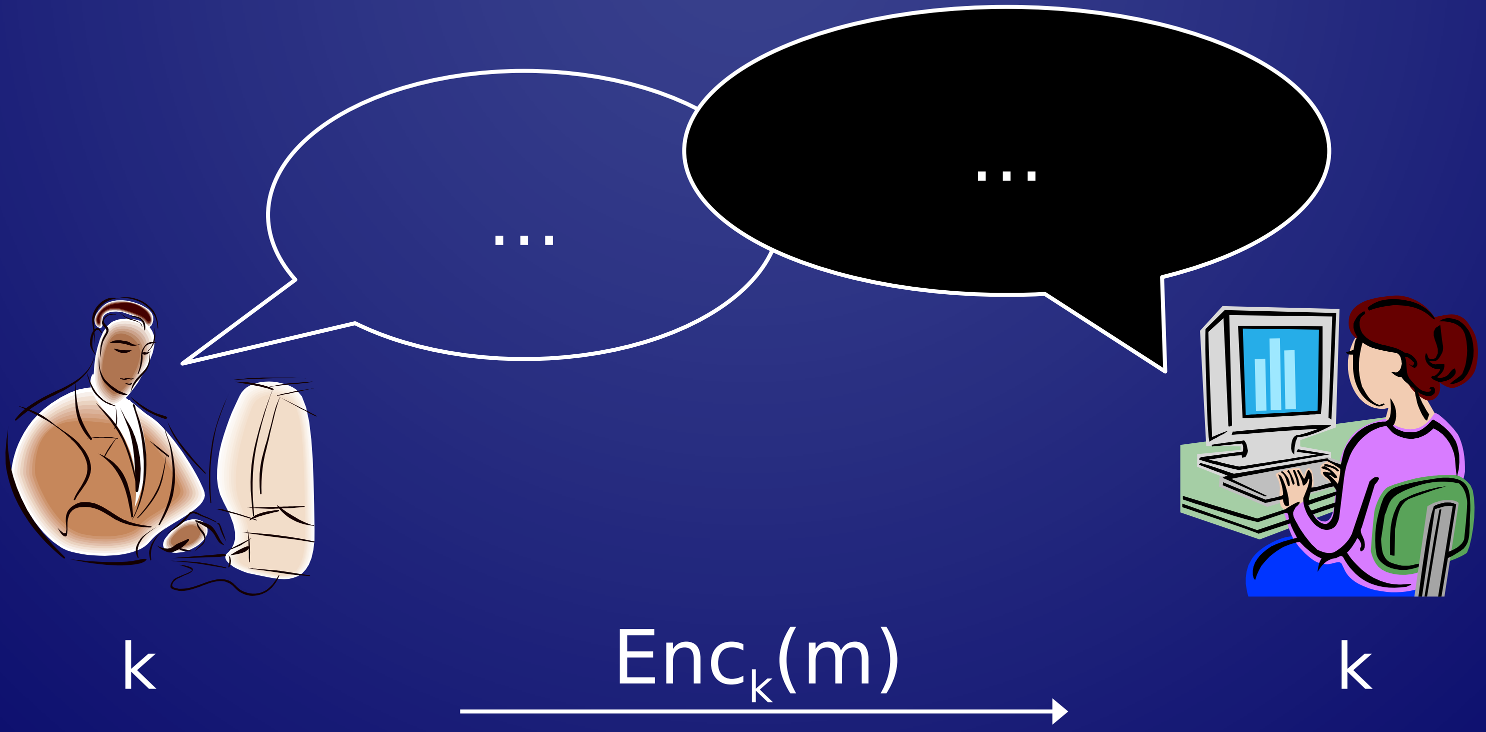
Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user of the system to send a message to any other user enci-



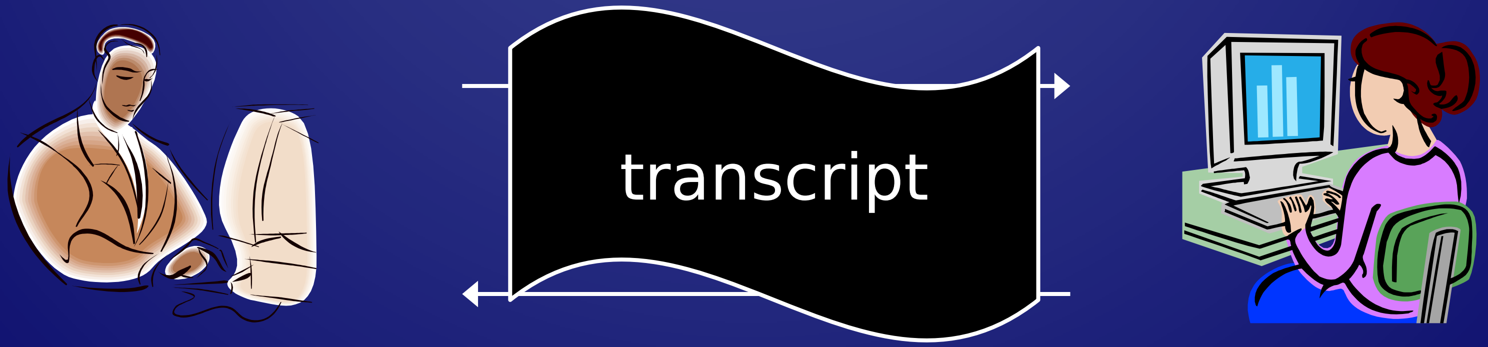
NEW DIRECTIONS…

- Key ideas:
  - Some problems exhibit *asymmetry* – easy to compute, but hard to invert (think factoring)
  - Use this asymmetry to enable two parties to agree on a shared secret key using public discussion(!)
    - *Key exchange*

KEY EXCHANGE



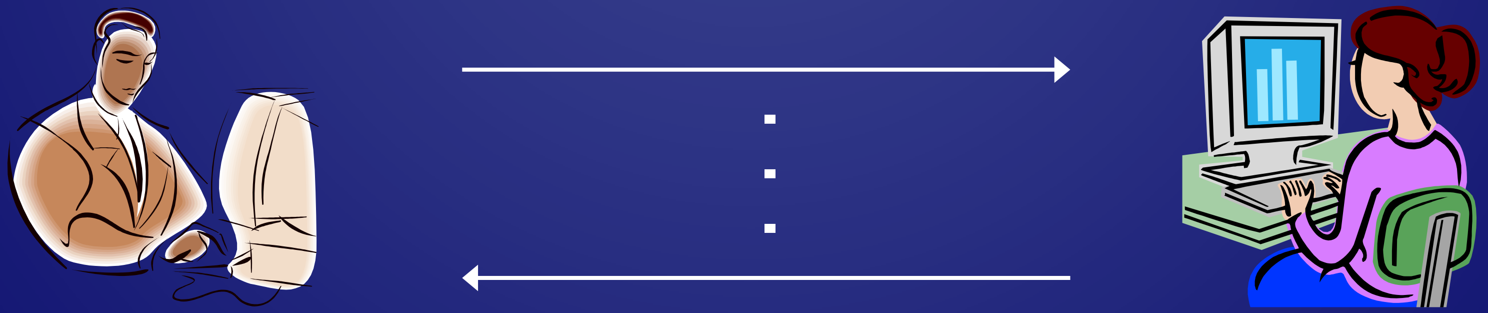
MORE FORMALLY…



$k \leftarrow \{0,1\}^n$ $k \leftarrow \{0,1\}^n$

<u>Security goal:</u> even after observing the transcript, the shared key k should be indistinguishable from a uniform key

KEY EXCHANGE



$k \leftarrow \{0,1\}^n$ $k \leftarrow \{0,1\}^n$

Security goal: even after observing the transcript, the shared key k should be indistinguishable from a uniform key

DIFFIE-HELLMAN KEY EXCHANGE

- Group-generation algorithm G outputs cyclic group G of prime order q with generator g
  - |q| = n bits
- *Decisional Diffie-Hellman (DDH) problem:*
  - Given g^x, g^y, distinguish g^{xy} from a uniform group element
- Hardness of DDH implies hardness of the discrete-logarithm problem
  - (This alone is not enough for key exchange)

DIFFIE-HELLMAN KEY EXCHANGE

$$\xrightarrow{G, q, g, h_1}$$

$$\xleftarrow{h_2}$$

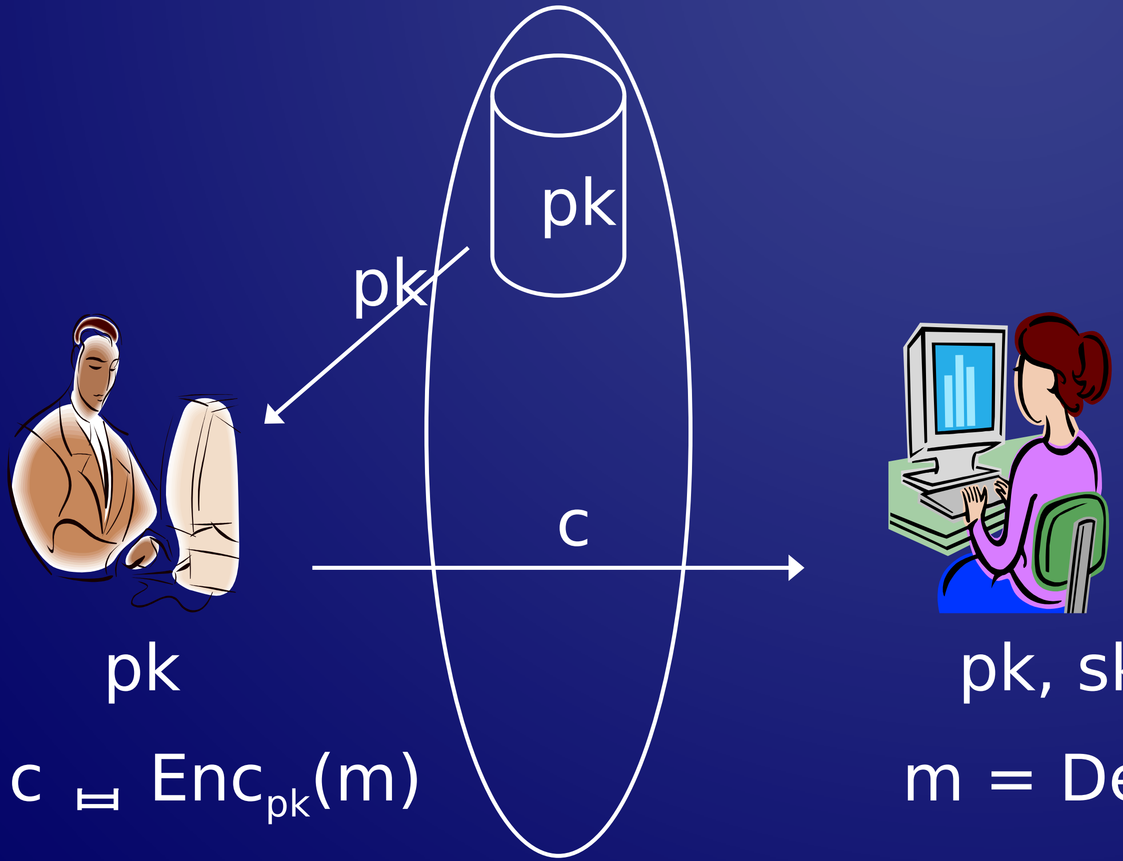
$k_1 = (h_2)^x = g^{yx}$

$k_2 = (h_1)^y = g^{xy}$

$(G, q, g) \leftarrow \mathcal{G}(1^n)$
$x \leftarrow \mathbb{Z}_q$
$h_1 = g^x$

$y \leftarrow \mathbb{Z}_q$
$h_2 = g^y$

PUBLIC-KEY ENCRYPTION



$c \leftarrow \mathrm{Enc}_{pk}(m)$

$m = \mathrm{Dec}_{sk}(c)$

EL GAMAL ENCRYPTION

Public key

$G, q, g, h_1 \longrightarrow$

$\longleftarrow h_2, h_1^y \cdot m$

$\longleftarrow c_2 = k \cdot m$

$(G, q, g) \leftarrow G(1^n)$

$x \leftarrow \mathbb{Z}_q$

$h_1 = g^x$

$k = (h_2)^x$

$m = c_2/k$

$y \leftarrow \mathbb{Z}_q$

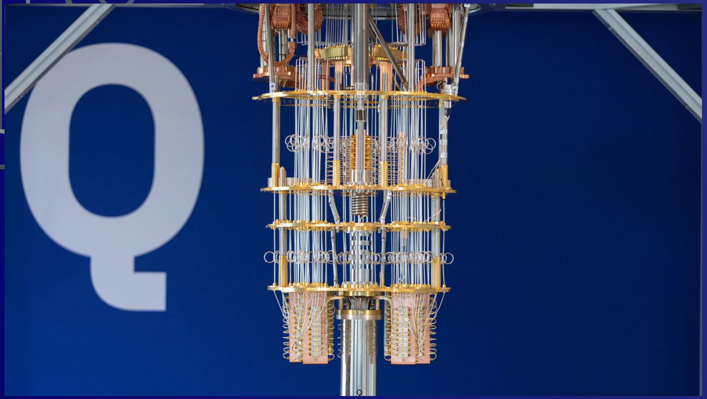
$h_2 = g^y$

$k = (h_1)^y$

EL GAMAL ENCRYPTION

- Gen(1^n)
  - Run G(1^n) to obtain G, q, g. Choose uniform $x \in \mathbb{Z}_{q}$. The public key is (G, q, g, g^x) and the private key is x
- Enc$_{pk}$(m), where pk = (G, q, g, h) and $m \in G$
  - Choose uniform $y \in \mathbb{Z}_{q}$. The ciphertext is g^y, $h^y \cdot m$
- Dec$_{sk}$(c_1, c_2)
  - Output $c_2/c_1^{\,x}$

QUANTUM COMPUTERS



- GOOGLE Corporation, in conjunction with with the company D-Wave signed contract about creating quantum computers. D-Wave 2X - is the newest quantum processor, which contains physical qubits.
- Each additional qubit doubles the data search area, thus is also significantly increased the calculation speed. Quantum computers will destroy systems based on the problem of factoring integers (e.g., RSA). RSA cryptosystem is used in different products on different platforms and in different areas.

RSA system is widely used in operating systems from Microsoft, Apple, Sun, and Novell. In hardware performance RSA algorithm is used in secure phones, Ethernet, network cards, smart cards, and is also widely used in the cryptographic hardware. Along with this, the algorithm is a part of the underlying protocols protected Internet communications, including S / MIME, SSL and S / WAN, and is also used in many organizations, for example, government, banks, most corporations, public laboratories and universities.

NEWS FROM GOOGLE

- Google made a huge revelation on October 23, 2019, when it announced that it had reached something called “quantum supremacy.” Via an article in the journal Nature, Google said their quantum computer, called Sycamore, solved a particularly difficult problem in 200 seconds. For comparison, Google said the world’s current fastest classical computer — one called Summit owned by IBM that’s as big as two basketball courts — would take 10,000 years to solve that same problem. This is what “quantum supremacy” means. It’s when a quantum computer — one that runs on the laws of quantum physics as opposed to the classical computers we’re familiar with (i.e. phones and laptops), which run on classical physics like Newton’s laws of motion — does something that no conventional computer could do in a reasonable amount of time.

IBM’S ANSWER

- IBM responded to Google’s news to say that actually, Summit could solve the quantum computers’ problem in two and a half days — not 10,000 years as Google had suggested. In this episode of Recode’s Reset podcast, host Arielle Duhaime-Ross and Kevin Hartnett, a senior writer for the math and physics magazine Quanta, break down exactly what quantum computing is and why Google dunking on IBM both was and wasn’t a huge deal.

CHINESE RESEARCHERS ACHIEVE QUANTUM ADVANTAGE IN TWO MAINSTREAM ROUTES

- Chinese research teams have made marked progress in superconducting quantum computing and photonics quantum computing technology, making China the only country to achieve quantum computational advantage in two mainstream technical routes, while the US has only achieved a "quantum advantage" in superconducting quantum computing, analysts say.
- "Zuchongzhi 2.1," is 10 million times faster than the current fastest supercomputer and its calculation complexity is more than 1 million times higher than Google's Sycamore processor. It's the first time that China has reached quantum advantage in a superconducting quantum computing system.

- Pan's team also built a new light-based quantum computer prototype, "Jiuzhang 2.0," with 113 detected photons, which can implement large-scale Gaussian boson sampling (GBS) 1 septillion times faster than the world's fastest existing supercomputer, according to the Xinhua News Agency.

  Yuan said that the number of detected photons for "Jiuzhang 2.0" increased to 113 from the previous 76 when the quantum computer prototype "Jiuzhang" first came out, which was a major technical breakthrough, as the difficulty increases exponentially with each additional detected photon.

  The light-based quantum computer prototype "Jiuzhang" was built in December 2020, led by Pan and Lu, and demonstrated a quantum advantage.

RSA ALTERNATIVES

Hash-based Digital Signature Schemes: One of RSA alternatives are Hash-based Digital Signature Schemes. The safety of these systems depends on the security of cryptographic hash functions.

A code-based public-key encryption system: McEliece example. In this system the public key is (G_{new}, t), and the private key is (S, G, P), where G is k x n generator matrix for the code C. C is random binary (n, k)-linear code, that is capable to improve t errors. N is the number of code words, k is dimension of C. S is a random k x k binary nonsingular matrix. P is a random n x n binary permutation matrix. $G_{new} =$ S * G * P; k x n matrix. To encrypt the message we must encrypt message m as a binary string with the length k; cyp = m x G_{new}; is generated random n-bit error vector v with the weight t. The cypher is calculated as c= cyp+v. For decoding is calculated cyp = c*P^{-1}; Using decryption algorithm of C is calculated m_{new}= m*S => m= m_{new}*S^{-1}

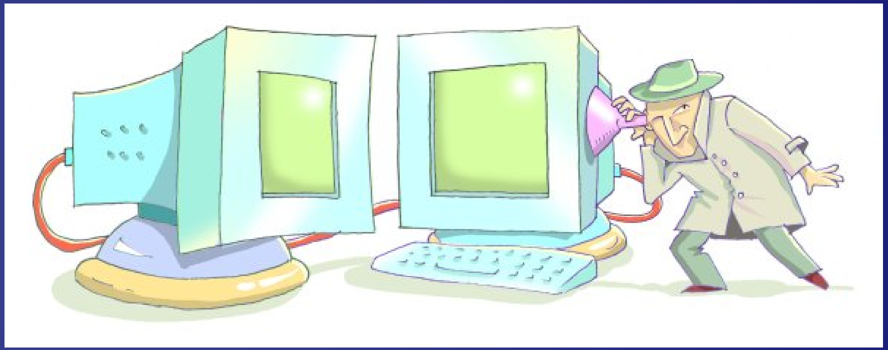
RSA ALTERNATIVES

- **Lattice-based Cryptography:** proofs are based on worst-case hardness.
- **Multivariate public key cryptosystem – MPKCs:** have a set of(usually) quadratic polynomials over a finite field. Security assumption is backed by the NP-hardness of the problem to solve nonlinear equations over a finite field.

SUCCESSFUL ATTACKS

- To date are already found successful attacks on this crypto system.
- The Ph.D. candidate of Dublin City University (DCU) Neill Costigan with the support of Irish Research Council for Science, Engineering and Technology (IRCSET), together with professor Michael Scott, Science Foundation Ireland (SFI) member successfully were able to carry out an attack on the algorithm. To do this they needed 8,000 hours of CPU time. In the attack representatives of four other countries took part. Scientists have discovered that the initial length of the key in this algorithm is insufficient and should be increased.
- This system cannot be also used to encrypt the same message twice and to encrypt the message when is known it’s relation with the other message.

- Should be noted the importance of efficiency spectrum. To date experts have reached quite good results in the speed algorithm processing. According to the investigation results it becomes clear that the proposed post-quantum cryptosystems are relatively little effective. Implementation of the algorithms requires much more time for their processing and verification.
- Inefficient cryptography may be acceptable for the general user, but it cannot be acceptable for the internet servers that handle thousands of customers in the second. Today, Google has already has problems with the current cryptography. It is easy to imagine what will happen when implementing crypto algorithms will take more time.
- The development and improvement of modern cryptosystems will take years. Moreover, all the time are recorded successful attacks on them. When is determined the encryption function, and it becomes standard, it needs the



- During the implementation it is necessary to ensure not only correct work of the function and the speed of its efficiency, but also to prevent any kind of leaks. Recently have been recorded successful «cache-timing» attacks on RSA and AES system, as a result of that Intel has added the AES instructions to its processors.
- McEliece system is vulnerable to attacks, related to side channel attacks. Was shown the successful timing attack on Patterson algorithm. This attack does not detect the key, but detects an error vector that can successfully decrypt the message cipher.
- As we can see, for the creation and implementation of safe and effective post-quantum cryptosystems it is necessary to fulfill the rather big work. From the foregoing it is clear that today we are not ready to transfer cryptosystems into post-quantum era. In the near future we cannot be sure in the reliability of the systems.

RSA ALTERNATIVES – HASH BASED

- Traditional digital signature systems that are used in practice are vulnerable to quantum computers attacks. The security of these systems is based on the problem of factoring large numbers and calculating discrete logarithms. Scientists are working on the development of alternatives to RSA, which are protected from attacks by quantum computer. One of the alternatives are hash based digital signature schemes. These systems use a cryptographic hash function. The security of these digital signature systems is based on the collision resistance of the hash functions that they use.

LAMPORT–DIFFIE ONE-TIME SIGNATURE SCHEME (KEY GENERATION)

- Keys generation in this system occurs as follows: the signature key X of this system consists of 2n lines of length n, and is selected randomly.
- $X= (x_{n-1}[0], x_{n-1}[1], \ldots, x_0[0], x_0[1]) \in \{0,1\}^{n,2n}$
- Verification key Y of this system consists of 2n lines of length n.
- $Y= (y_{n-1}[0], y_{n-1}[1], \ldots, y_0[0], y_0[1]) \in \{0,1\}^{n,2n}$
- This key is calculated as follows:
- $y_i[j] = f(x_i[j]), 0<=i<=n-1, j=0,1$
- f – is one-way function:
- f: $\{0,1\}^{n}$ ▭ $\{0,1\}^{n}$;

DOCUMENT SIGNATURE

- To sign a message m of arbitrary size, we transform it into size n using the hash function:
- $h(m)=hash = (hash_{n-1}, \dots , hash_0)$
- Function h- is a cryptographic hash function:
- $h: \{0,1\}^{*} \to \{0,1\}^{n}$
- The signature is done as follows:
- $sig= (x_{n-1}[hash_{n-1}], \dots, x_0[hash_0]) \in \{0,1\}^{n,n}$
- i-th string in this signature is equals to $x_i[0]$, if i-th bit in hashed message is equal to 0. The string is equal to $x_i[1]$, if i-th bit in sign is equal to 1.
- Signature length is **n^2**.

DOCUMENT VERIFICATION

To verify the signature sig = (sig_{n-1}, …, sig_0), is calculated hash of the message hash = ($hash_{n-1}$, … , $hash_0$) and the following equality is checked:

$(f(sig_{n-1}), \ldots, f(sig_0)) = (y_{n-1}[hash_{n-1}], \ldots, y_0[hash_0])$

If the equation is true, then the signature is correct.

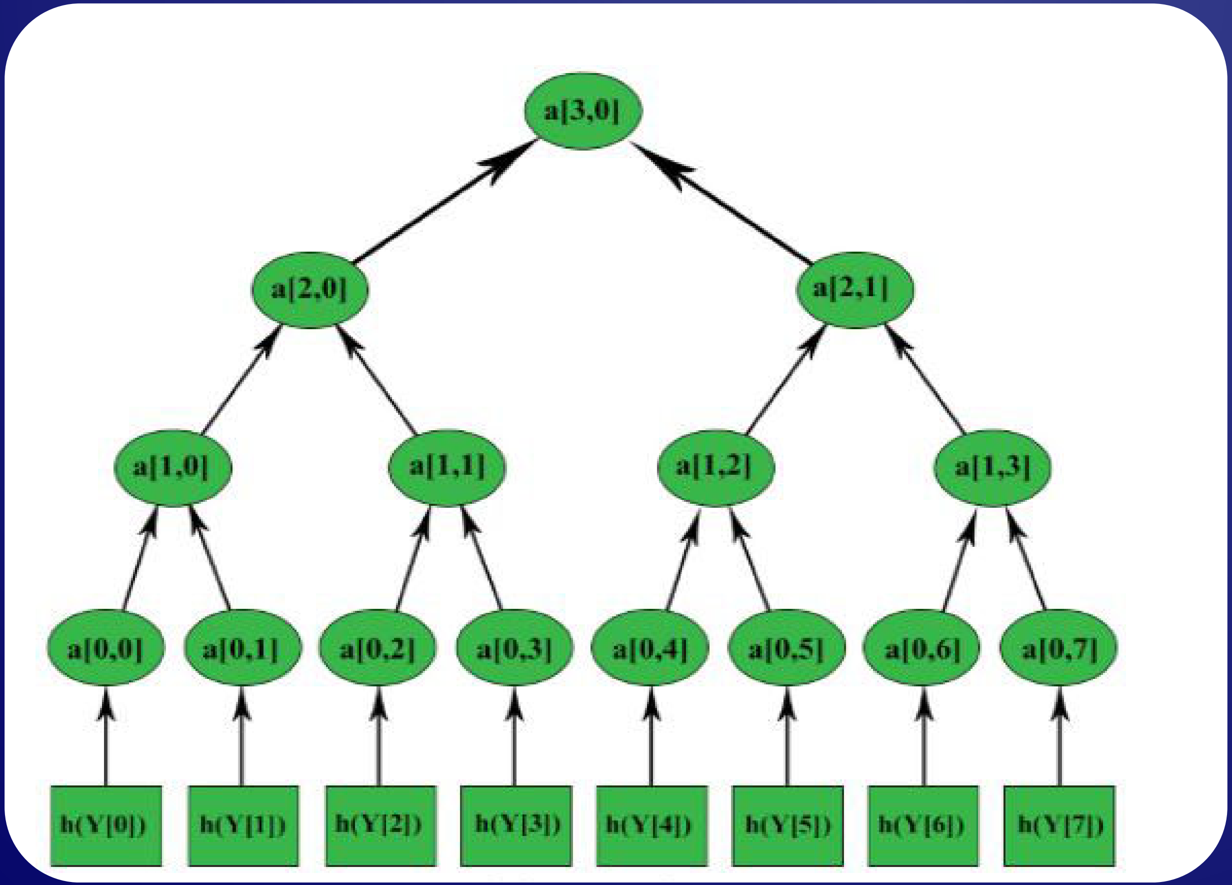
WINTERNITZ ONE TIME SIGNATURE SCHEME.
KEY GENERATION

To achieve security $O(2^{80})$, the total size of public and private keys must be $160*2*160$ bits = 51200 bits, that is 51200/1024=50 times larger than in the case of RSA. We must also note that the size of the signature in the given scheme is much larger than in the case of RSA. Winternitz One-time Signature Scheme was proposed to reduce the size of the signature.

MERKLE

- One-time signature schemes are very inconvenient to use, because to sign each message, you need to use a different key pair. Merkle crypto-system was proposed to solve this problem. This system uses a binary tree to replace a large number of verification keys with one public key, the root of a binary tree. This cryptosystem uses an one-time Lamport or Winternitz signature scheme and a cryptographic hash function:
- $h:\{0,1\}^* \rightarrow \{0,1\}^n$
- **Key generation:** The length of the tree is chosen $H>=2$, with one public key it is possible to sign 2^H documents. 2^H signature and verification key pairs are generated; X_i, Y_i, $0<=i<=2^H$. X_i- is signature key**,** Y_i- is verification key. $h(Y_i)$ are calculated and are used as the leaves of the tree. Each tree node is a hash value of concatenation of its children.

MERKLE TREE


a[3,0]
a[2,0]
a[2,1]
a[1,0]
a[1,1]
a[1,2]
a[1,3]
a[0,0]
a[0,1]
a[0,2]
a[0,3]
a[0,4]
a[0,5]
a[0,6]
a[0,7]
h(Y[0])
h(Y[1])
h(Y[2])
h(Y[3])
h(Y[4])
h(Y[5])
h(Y[6])
h(Y[7])

SIGNATURE GENERATION

- To sign a message m of arbitrary size we transform it into size n using the hash function
- h (m) = hash, and generate an one-time signature using any one-time key X_{any}, the document's signature will be the concatenation of: one time signature, one-time verification key Y_{any}, index any and all fraternal nodes $auth_i$ in relation to Y_{any}.
- Signature= $(sig||pub||any|| Y_{any}||auth_0,\dots,auth_{H-1})$
- **Signature verification:**
- To verify the signature we check the one-time signature of sig using Y_{any}, if it is true, we calculate all the nodes a [i, j] using "$auth_{i}$", index "any" and Y_{any}. We compare the last node, the root of the tree with public key, if they are equal, then the signature is correct.

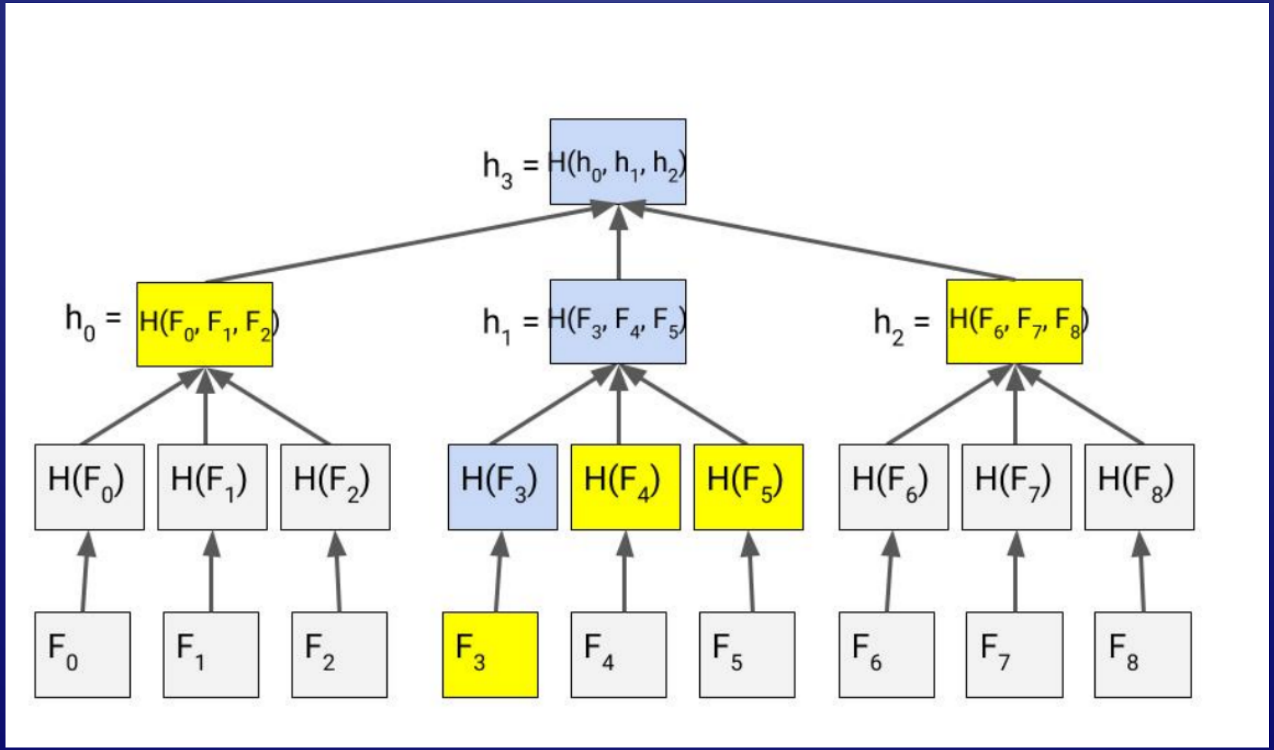
MERKLE TREE

- Merkle Trees are computationally fast, and a Merkle Tree over n nodes can be constructed in O(n) time.
- Merkle Tree that contains many nodes can have Merkle proofs that are then prohibitively large.
- To sign 2^n messages the height of the tree must be n.
- The Merkle Proof itself could create a large and expensive bandwidth overhead on Dropbox

K-ARY MERKLE TREES

- One possible solution is to use a k-ary Merkle Tree. In a binary Merkle Tree, the proof consists of one node at each level, so to reduce the size of the proof, we can reduce the height of the tree by giving it a branching factor of $k > 2$.
- This approach reduces the height of the tree, but enlarges the proof size. If a branching factor is k, it reduces the height of the tree from $\log_2 n$ to $\log_k n$. $\log_2 k$ is decrease in height.
- Merkle proof actually grows larger, from $O(\log_2 n)$ to $O(k \log_k n)$.

K-ARY MERTKLE TREE


h_3 = H(h_0, h_1, h_2)
h_0 = H(F_0, F_1, F_2)
h_1 = H(F_3, F_4, F_5)
h_2 = H(F_6, F_7, F_8)
H(F_0)
H(F_1)
H(F_2)
H(F_3)
H(F_4)
H(F_5)
H(F_6)
H(F_7)
H(F_8)
F_0
F_1
F_2
F_3
F_4
F_5
F_6
F_7
F_8

OUR GOAL

- Our goal is reduce:
- 1. the height
- 2. the proof size

VECTOR COMMITMENT TREE

- In Merkle Tree, we replace the Hash functions with the corresponding Vector Commitments.
- To compute a Verkle Tree for the messages, $m_0,m_1,\ldots,m_n$:

1. The branching factor of the tree is selected, **k**.
2. We group our messages into subsets of k and calculate a Vector Commitment, **VC**, over each of the subsets.
3. We compute each membership proofs $\mathbf{p_i}$ for every message m_i in the subset with respect to VC.
4. After we continue computing Vector Commitments up the tree over previously computed commitments until we compute the root commitment of Verkle tree.

VERKLE TREE

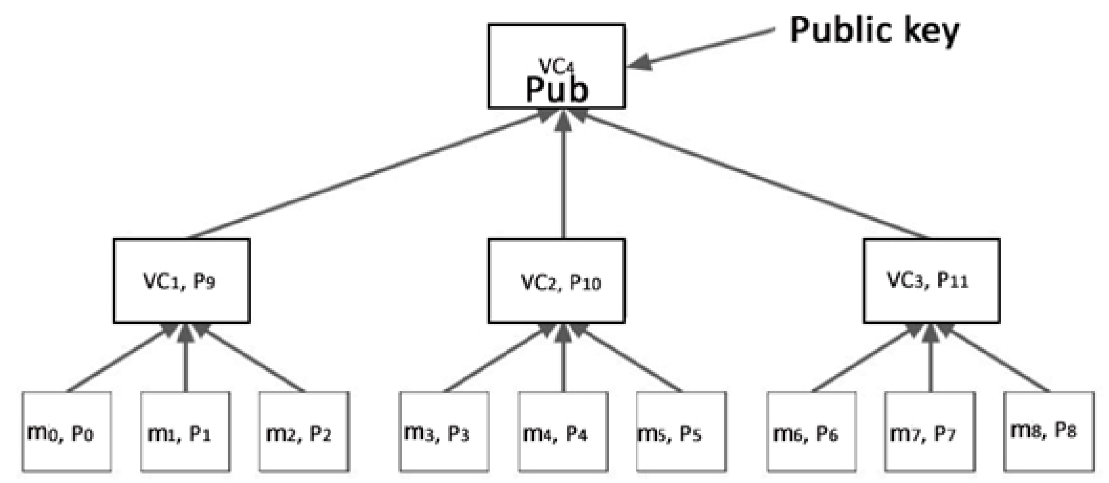


Fig 2. Verkle Signature Scheme

COMPLEXITY

Scheme	Construction	Proof size
Merkle Tree	O(n)	$O(\log_2 n)$
k-ary Merkle Tree	O(n)	$O(k \log_k n)$
Verkle Tree	$O(n^2)$	O(1)
k-ary Verkle Tree	O(kn)	$O(\log_k n)$

VECTOR COMMITMENTS

- Vector commitment allows to commit to an ordered sequence of values in such a way that it is later possible to open the commitment only w.r.t. a specific position. We define Vector Commitments as a non-interactive primitive.

ALGORITHMS

- Vector commitments can be described via the following algorithms:
- VC.KeyGen (1^k ,q) Given the security parameter k and the size q of the committed vector (with q = poly (k)), the key generation outputs some public parameters pp (which implicitly define the message space M).
- VC . Com$_{pp}$ (m$_1$,...,m$_q$) On input a sequence of q messages m$_1$,...,m$_q$ ∈ M and the public parameters pp , the committing algorithm outputs a commitment string C and an auxiliary information aux .
- VC.Open$_{pp}$ (m,i,aux) This algorithm is run by the committer to produce a proof Λ$_i$ that m is the i - th committed message.
- VC.Ver$_{pp}$ (C,m,i,Λ$_i$) The verification algorithm accepts (i.e., it outputs 1) onlyif Λ$_i$ is a valid proof that C was created to a sequence m$_1$,...,m$_q$ such that m = m$_i$.

VECTOR COMMITMENTS

- 1. Vector Commitment Based on CDH
- 2. Vector Commitment Based on RSA

VECTOR COMMITMENT BASED ON CDH

- Here we propose an implementation of concise vector commitments based on the CDH assumption in bilinear groups. Precisely, the security of the scheme reduces to the Square Computational Diffie-Hellman assumption. Roughly speaking, the Square-CDH assumption says that it is computationally infeasible to compute the value g^(a^2), given g,g^a ∈ G.

VECTOR COMMITMENT BASED ON CDH ALGORITHMS

- VC.KeyGen (1^k ,q) Let G , G_T be two bilinear groups of prime order p equipped with a bilinear map e : G × G → G_T . Let g ∈ G be a random generator.Randomly choose z_1 ,...,z_q from Z p .
- For all i = 1 ,...,q set: h_i = g^(z_i) . For all i,j = 1 ,...,q , i≠j set $h_{i,j}$ = g^(z_iz_j) Set pp = (g, $\{h_i\}_{i\in[q]}$, $\{ h_{i,j}\}_{i,j\in[q] ,i\neq j}$). The message space is M = Z_p
- VC.Com$_{pp}$ (m_1 ,...,m_q) Compute C = h_1^(m_1)h_2^(m_2) ··· h_q ^(m_q) and output C and the auxiliary information aux = (m1 ,...,m_q).
- VC.Open$_{pp}$ (m_i ,i, aux) Compute

$$\Lambda_i = \prod_{j=1,j\neq i}^{q} h_{i,j}^{m_j} = \left(\prod_{j=1,j\neq i}^{q} h_j^{m_j} \right)^{z_i}$$

- VC.Ver$_{pp}$ (C,mi ,i,Λi) If e(C/hi^(mi),hi)=e(Λi ,g) then output 1. Otherwise output 0.

POLYNOMIAL COMMITMENTS

- In practice, we use more powerful primitive than a vector commitment - a **polynomial commitment**. Polynomial commitments let you hash a polynomial, and make a proof for the evaluation of the hashed polynomial at *any* point. You can use polynomial commitments as vector commitments: if we agree on a set of standardized coordinates $(c_1,c_2...c_n)$, given a list $(y_1,y_2...y_n)$ you can commit to the polynomial P where
- $P(c_i)=y_i$ for all $i\in[1..n]$ you can find this polynomial with Lagrange interpolation.
- In numerical analysis, the **Lagrange interpolating polynomial** is the unique polynomial of lowest degree that interpolates a given set of data.

POLYNOMIAL COMMITMENTS

- The two polynomial commitment schemes that are the easiest to use are KZG commitments and bulletproof-style commitments (in both cases, a commitment is a single 32-48 byte elliptic curve point). Polynomial commitments give us more flexibility that lets us improve efficiency, and it just so happens that the simplest and most efficient vector commitments available *are* the polynomial commitments.
- This scheme is already very powerful as it is: **if you use a KZG commitment and proof, the proof size is 96 bytes per intermediate node, nearly 3x more space-efficient than a simple Merkle proof** if we set width = 256.

MERGING THE PROOFS

- Instead of requiring one proof for each commitment along the path, by using the extra properties of polynomial commitments we can make a single fixed-size proof that proves all parent-child links between commitments along the paths for an unlimited number of keys.

PROBLEMS

- Vector commitments based on CDH and RSA can be broken by quantum computers
- Polynomial commitments based on elliptic curves can be broken by quantum computers

VECTOR COMMITMENTS FROM LATTICES

- Chris Peikert, Zachary Pepin and Chad Sharp offer statelessly updatable VCs.
- The scheme is post-quantum

NOVEL SCHEME

- **Key generation:** The tree length is selected as H>=2. Here one public key can sign 2H documents. 2H key pairs Xi, and Yi are generated, where Xi is the signature key and Yi the verification key, h (Yi) are computed and used as the leaves of the tree. Each node in the tree is a hash value of of its children's concatenation**.**
- a[1,0]=h(a[0,0] || a[0,1])
- The public key of the **Verkle** crypto scheme is the root commitment, to generate it $\mathbf{2^H}$ pairs of one time keys must be computed.
- **Signature generation:** Message m of arbitrary size, is transformed into size n by means of the hash function. h (m) = hash, and is generated a one-time signature using arbitrary one-time key X_{arb}, the document's signature will be the concatenation of: one-time signature, one-time verification key Y_{arb}, index arb the proof and the root commitment. Signature= (sig||arb|| Y_{arb}||proofs ||commitments || root commitment)
- **Signature verification:** In Verkle digital signature verifying signature is done as following, the one-time signature of sig should be verified using Y_{arb}, if it is true, the commitments VC [i] are verified. If the root of the tree is equal to root commitment, the signature is verified.

QUESTIONS?

MAKSIM IAVICH

SCIENTIFIC CYBER SECURITY ASSOCIATION ; CAUCASUS UNIVERSITY

T. +(995 595) 511355; E-mail: miavich@cu.edu.ge

Scientific& practical cyber security journal

www

SCIENTIFIC CYBER SECURITY ASSOCIATION

DEEPSEC

კავკასიის უნივერსიტეტი
CAUCASUS UNIVERSITY