Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Part One

- One-way Functions (Hash Functions)
- Diffie–Hellman Protocol
- RSA Protocol
- DSA Protocol
- Elliptic curves cryptography basics

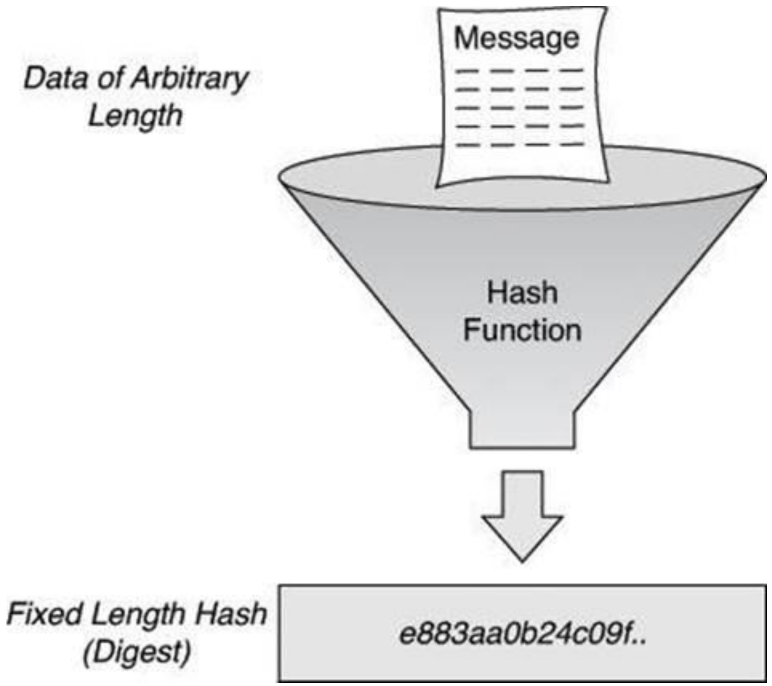
Part Two

- Schnorr signature algorithm
- ECDSA protocol
- BLS protocol
- NODR crypto-protocol (BLS-based)
- Beyond modern cryptography

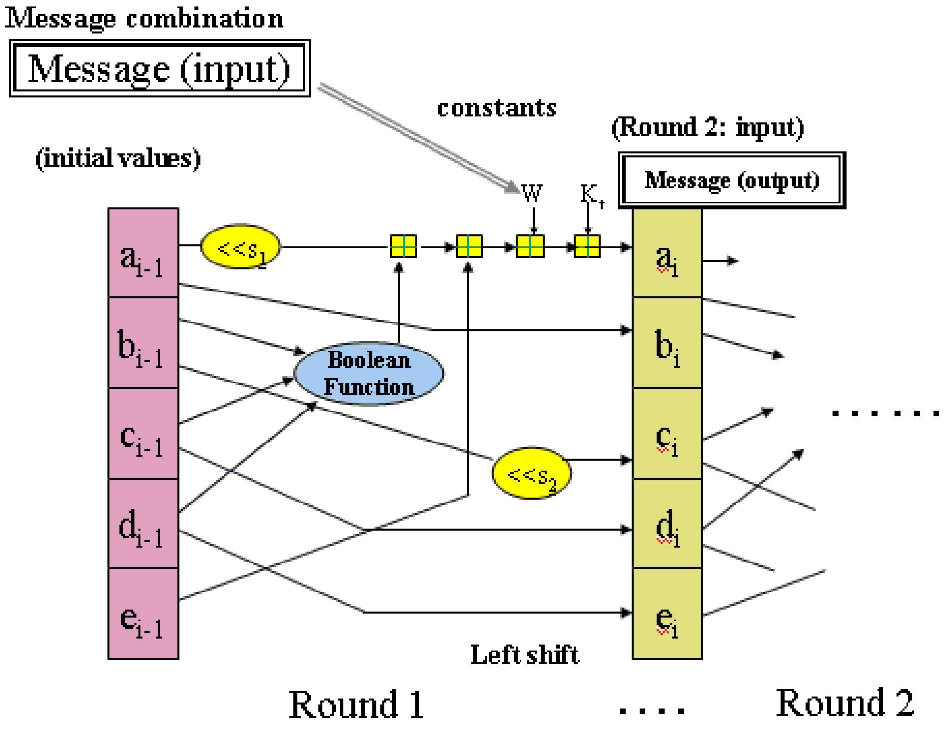
Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

One-way Functions (Hash Functions)



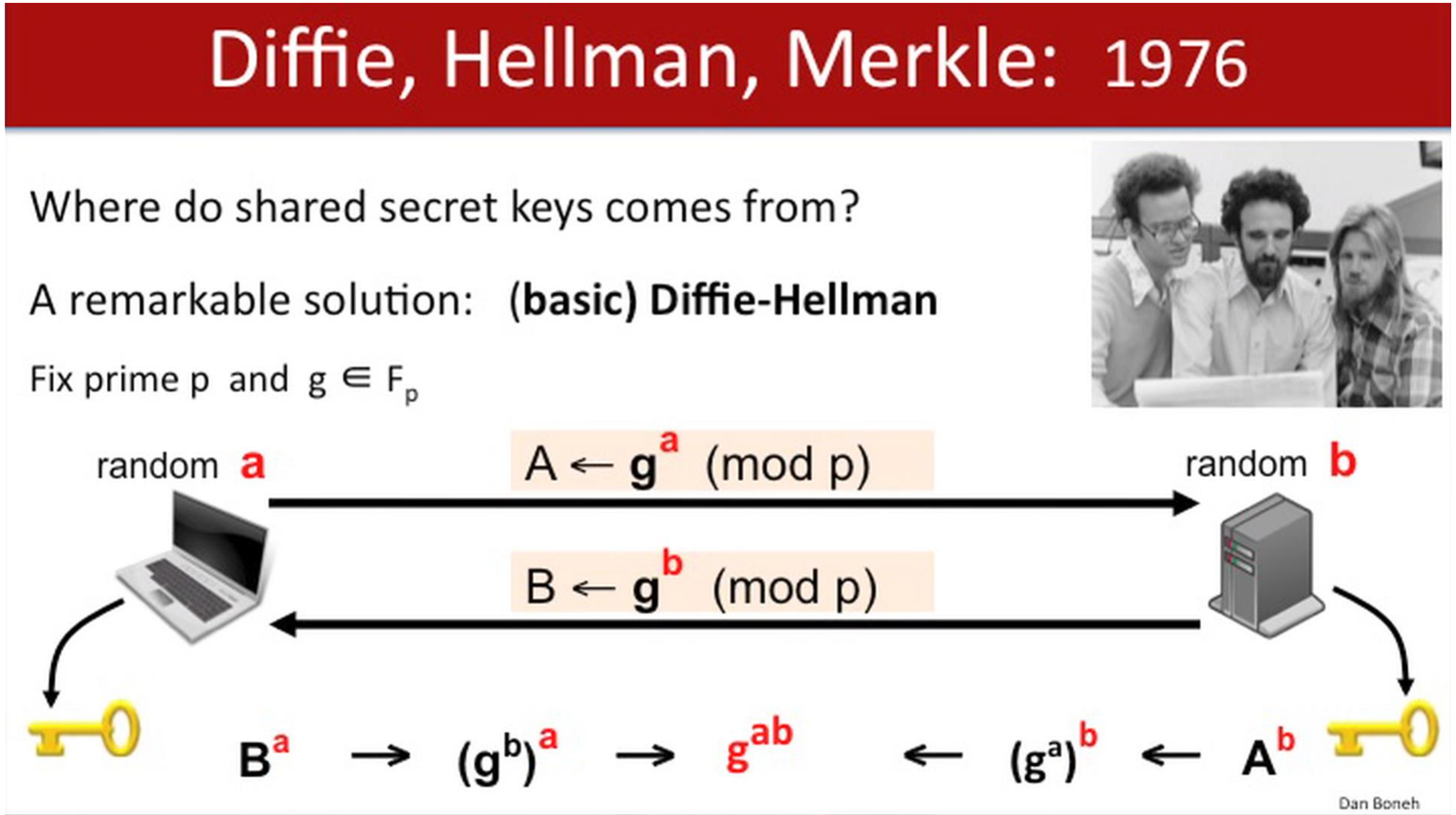
Principle of operation in a cryptographic hash function



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

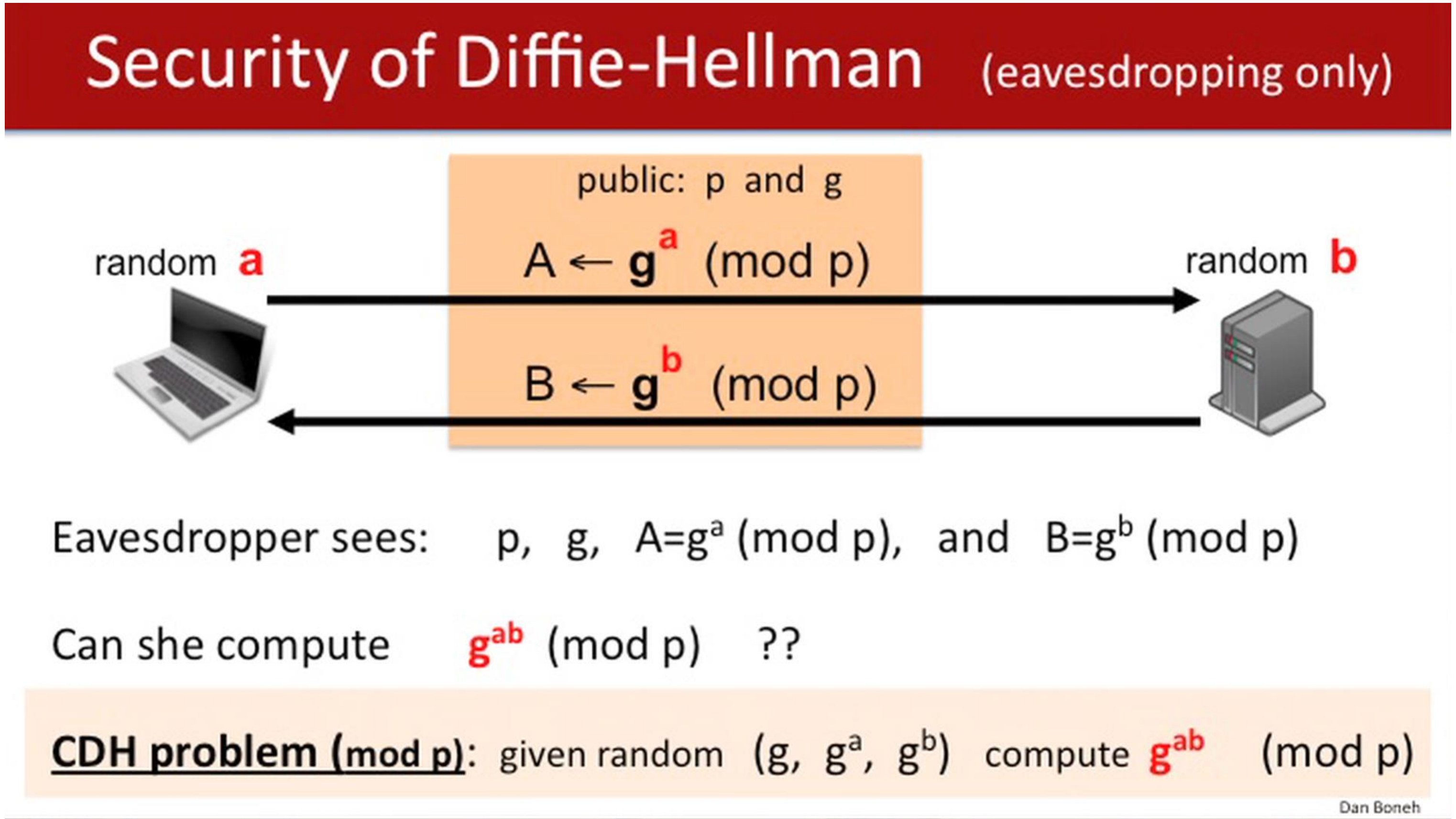
Diffie–Hellman Protocol



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Diffie–Hellman Protocol

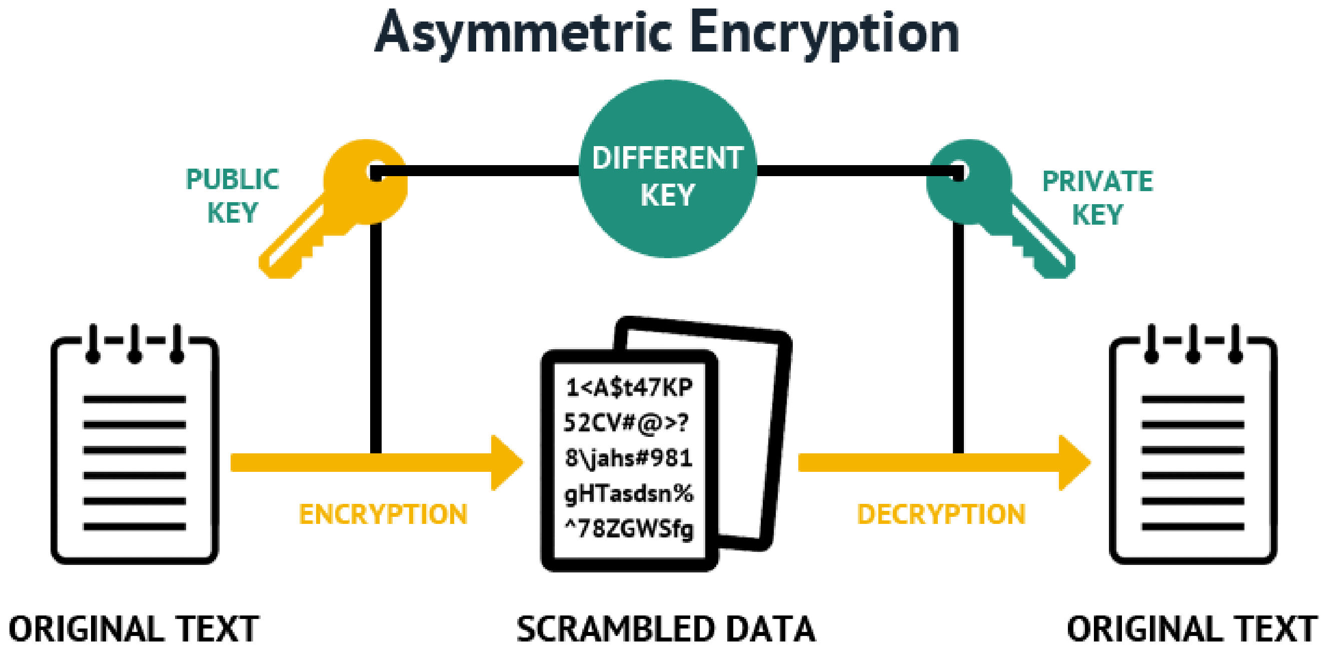


Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Diffie–Hellman Protocol

- Interactive (can we construct non-interactive?)
- Symmetric encryption (can we construct asymmetric?)



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

RSA Protocol (Rivest, Shamir, Adelman, 1977)

Generate key:

1. Generate two large unique prime numbers $\boldsymbol{p}$ and $\boldsymbol{q}$
2. Compute $n = p \times q$ and $\varphi = (p - 1) \times (q - 1)$
3. Select a random number $1 < e < \varphi$ such that $gcd(e, \varphi) = 1$
4. Compute the unique integer $1 < d < \varphi$ such that $e \times d \equiv 1 \ (mod\ \varphi)$
5. (d, n) is the private key
6. (e, n) is the public key

Encryption

1. Represent a message as an integer m in the interval [0, n-1]
2. Send out the encrypted data c

$$c = m^e \bmod n$$

Decryption:

1. Decrypt the key using

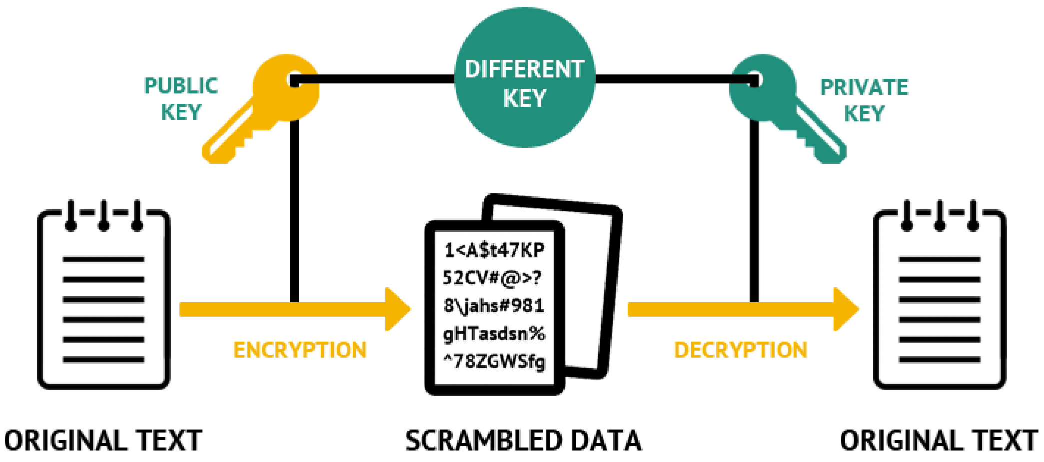
$$m = c^d \bmod n$$

Perm Summer School on Blockchain & Cryptomarkets 2019

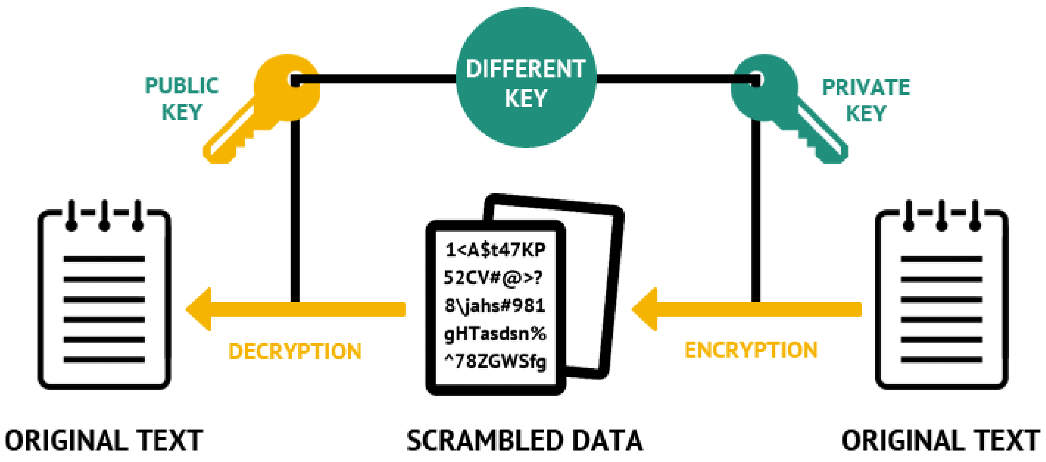
Survey on a modern cryptography workshop

DSA Protocol

Encrypted messaging



Digital signature



Encrypt $ciphertext = message^{e} \; mod \; n$

Decrypt $message = ciphertext^{d} \; mod \; n$

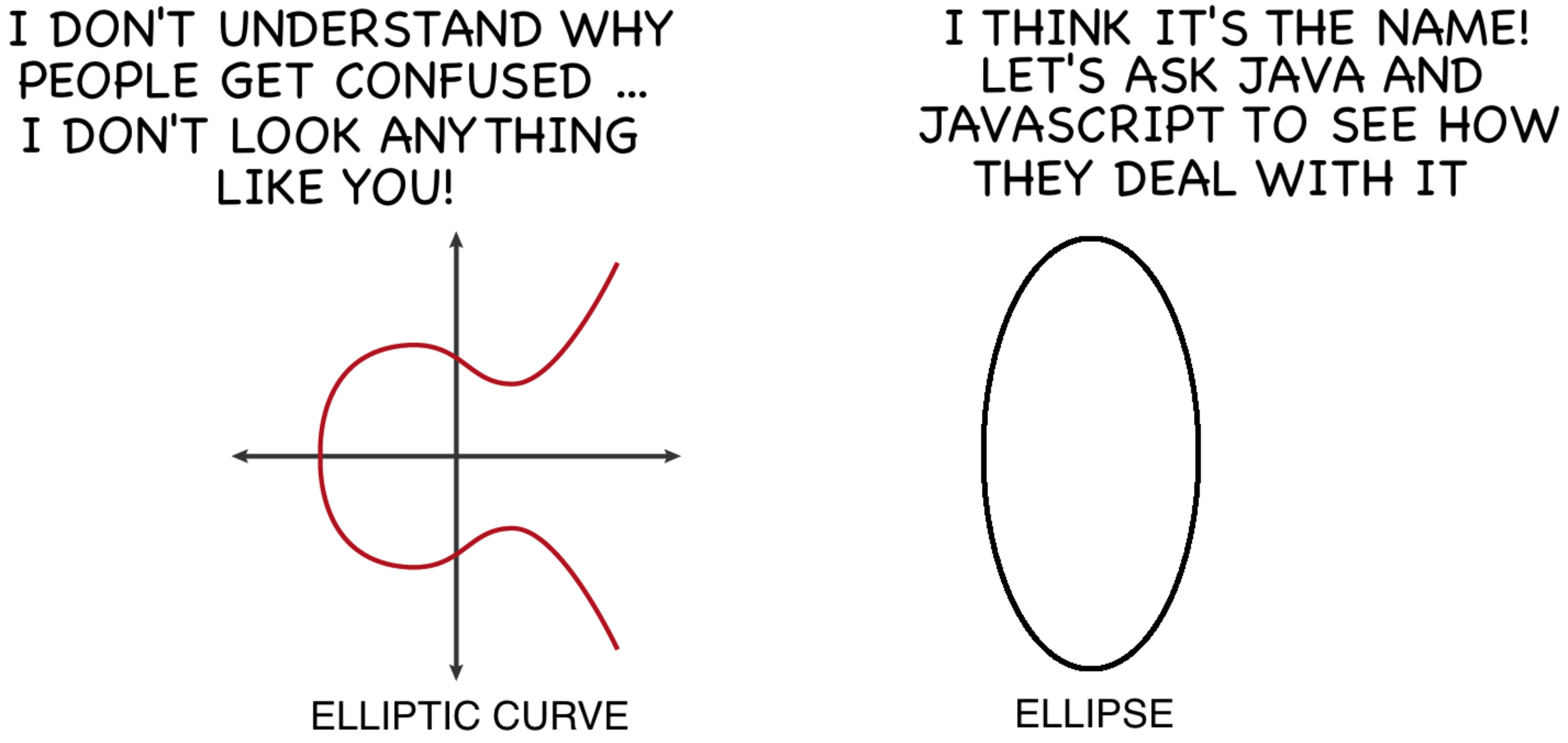
Sing $signature = message^{d} \; mod \; n$

Verify $message = signature^{e} \; mod \; n$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

Elliptic curve equation

$$y^2 = x^3 + ax + b$$

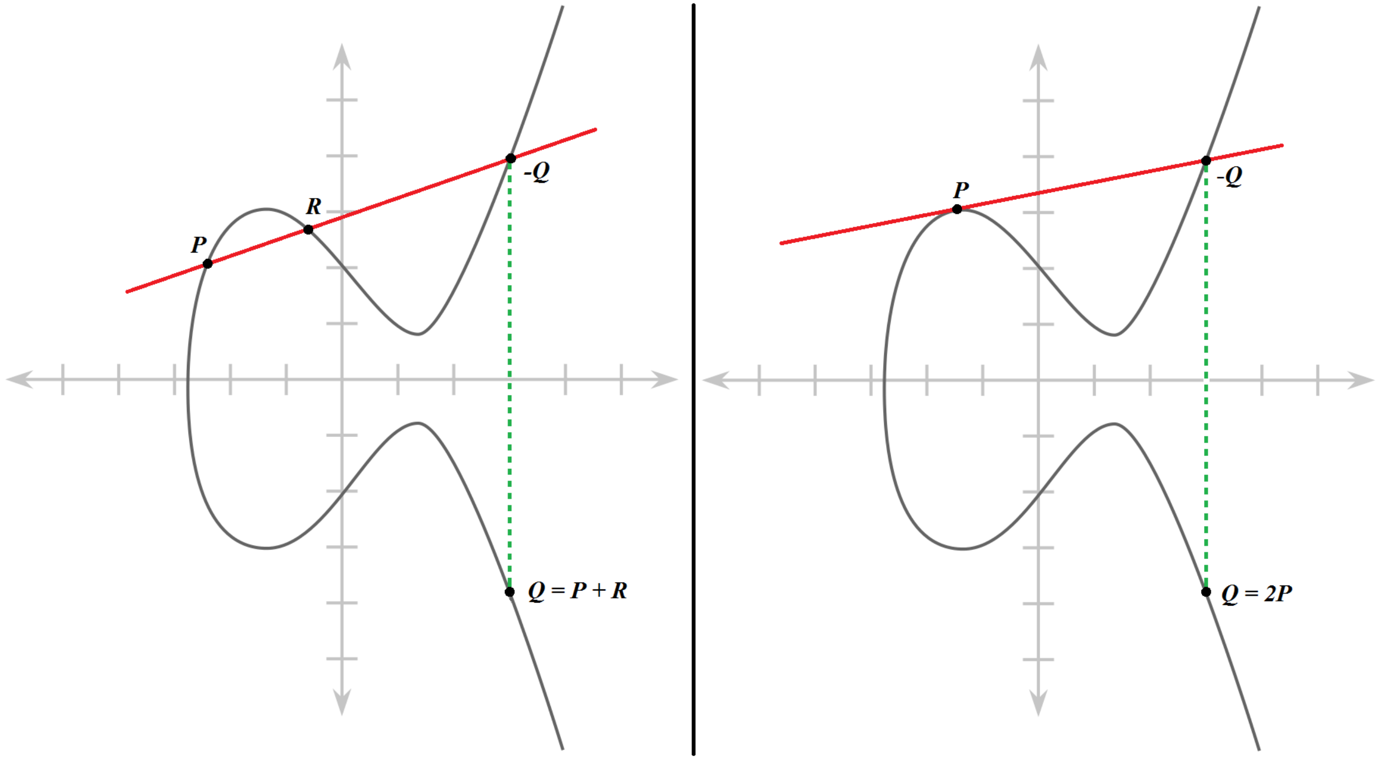
Point addition operation

$$P + R = Q \qquad Q - R = P$$

$$A + (B + C) = (A + B) + C$$

$$P + O = O + P = P$$

$$P + P = 2P$$
$$P + P + P = 3P$$
$$P + P + P + P = 4P$$



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

Addition of points $P_1=(x_1,y_1)$ and $P_2=(x_2,y_2)$ of an elliptic curve $E: y^2=x^3+ax+b$ can be easily computed using the following formulas:

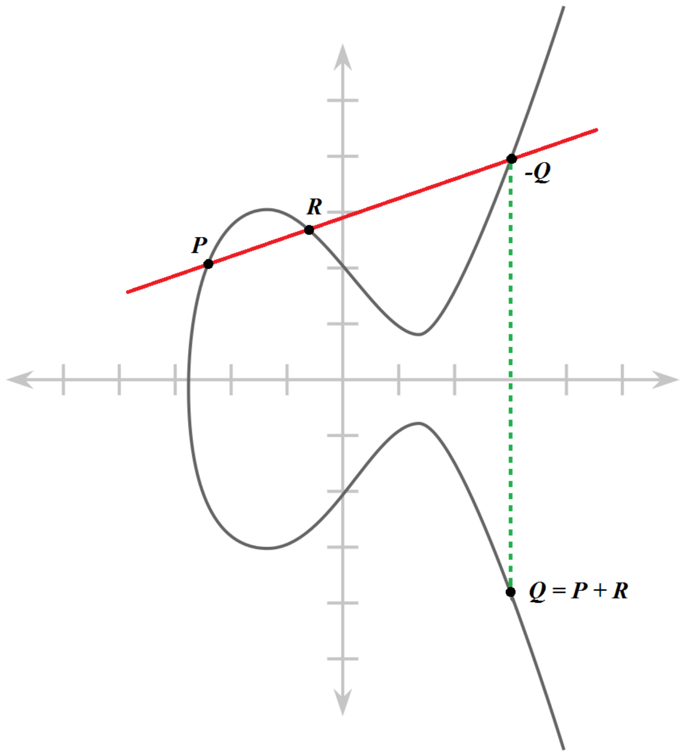
$$P_1 + P_2 = P_3=(x_3,y_3)$$

where

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = \lambda(x_1 - x_3) - y_1$$

and

$$\lambda = \begin{cases} (y_2-y_1)/(x_2-x_1) & \text{If } P_1 \neq P_2 \\ (3x_1^2+a)/(2y_1) & \text{If } P_1 = P_2 \end{cases}$$



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

Elliptic curve equation

$y^2 = x^3 + ax + b \quad \boldsymbol{mod\ p}$

Point addition operation

$P + R = Q \qquad Q - R = P$

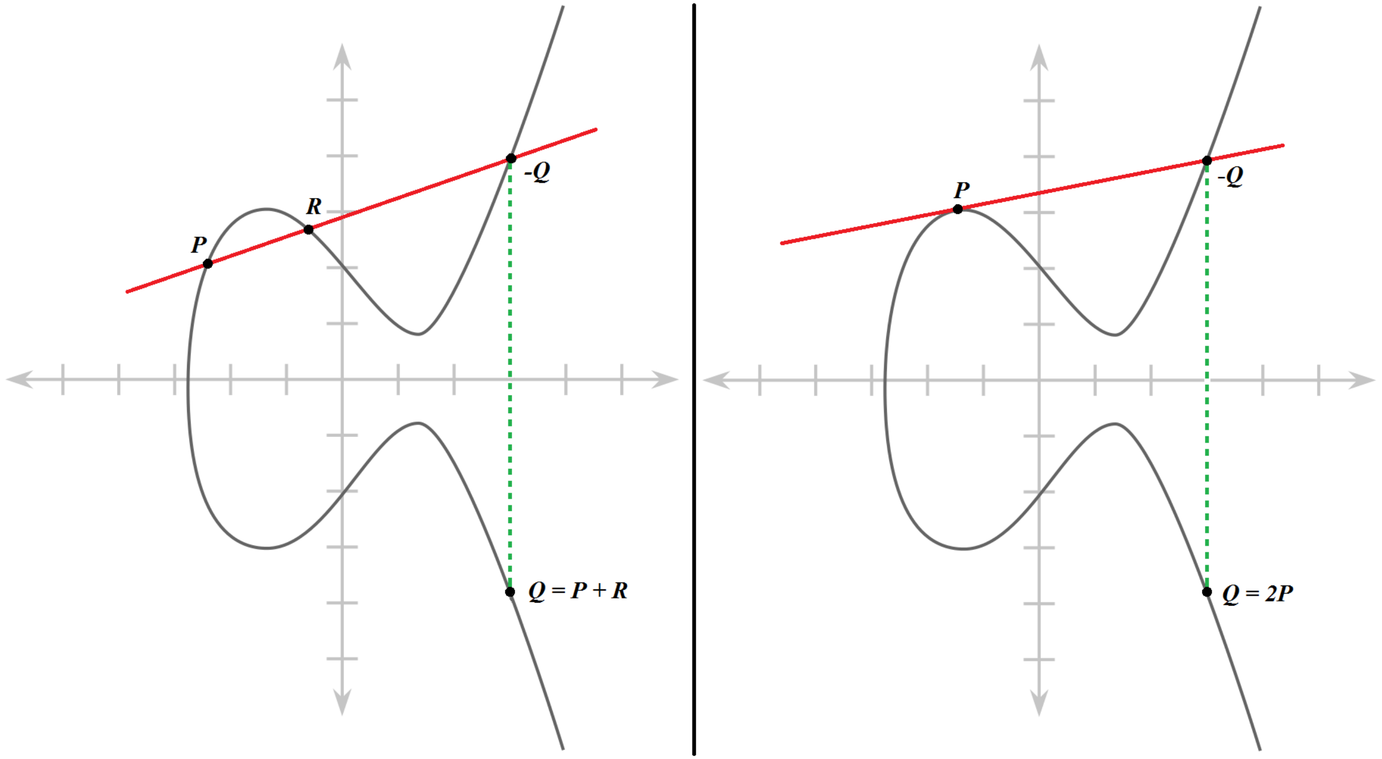
$A + (B + C) = (A + B) + C$

$P + O = O + P = P$

$P + P = 2P$

$P + P + P = 3P$

$P + P + P + P = 4P$



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

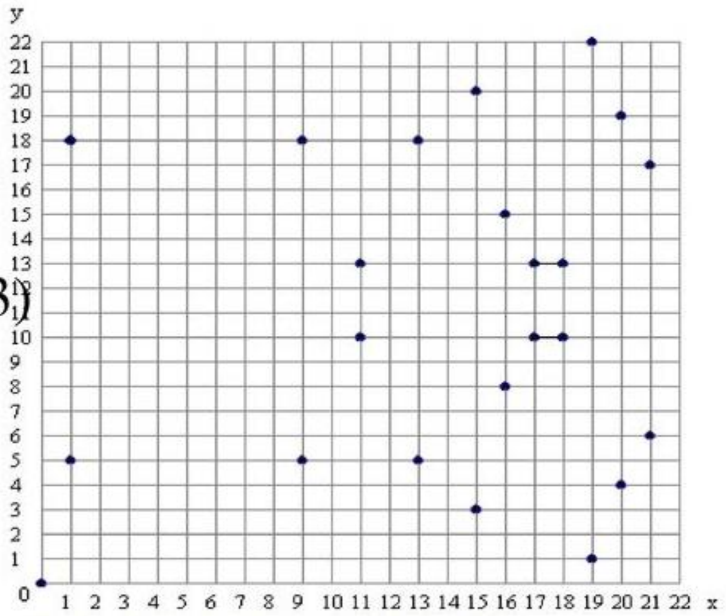
Elliptic curves cryptography basics

Set $p = 23$, $y^2 \bmod p = x^3 + 1x + 0 \bmod p$.

Choose G = (9,5) (on curve: 25 mod 23 = 729+9 mod 23)

The 23 points on this curve:

(0,0) (1,5) (1,18) (9,5) (9,18)
(11,10) (11,13) (13,5) (13,18)
(15,3) (15,20) (16,8) (16,15)
(17,10) (17,13) (18,10) (18,13)
(19,1) (19,22) (20,4) (20,19)
(21,6) (21,17)

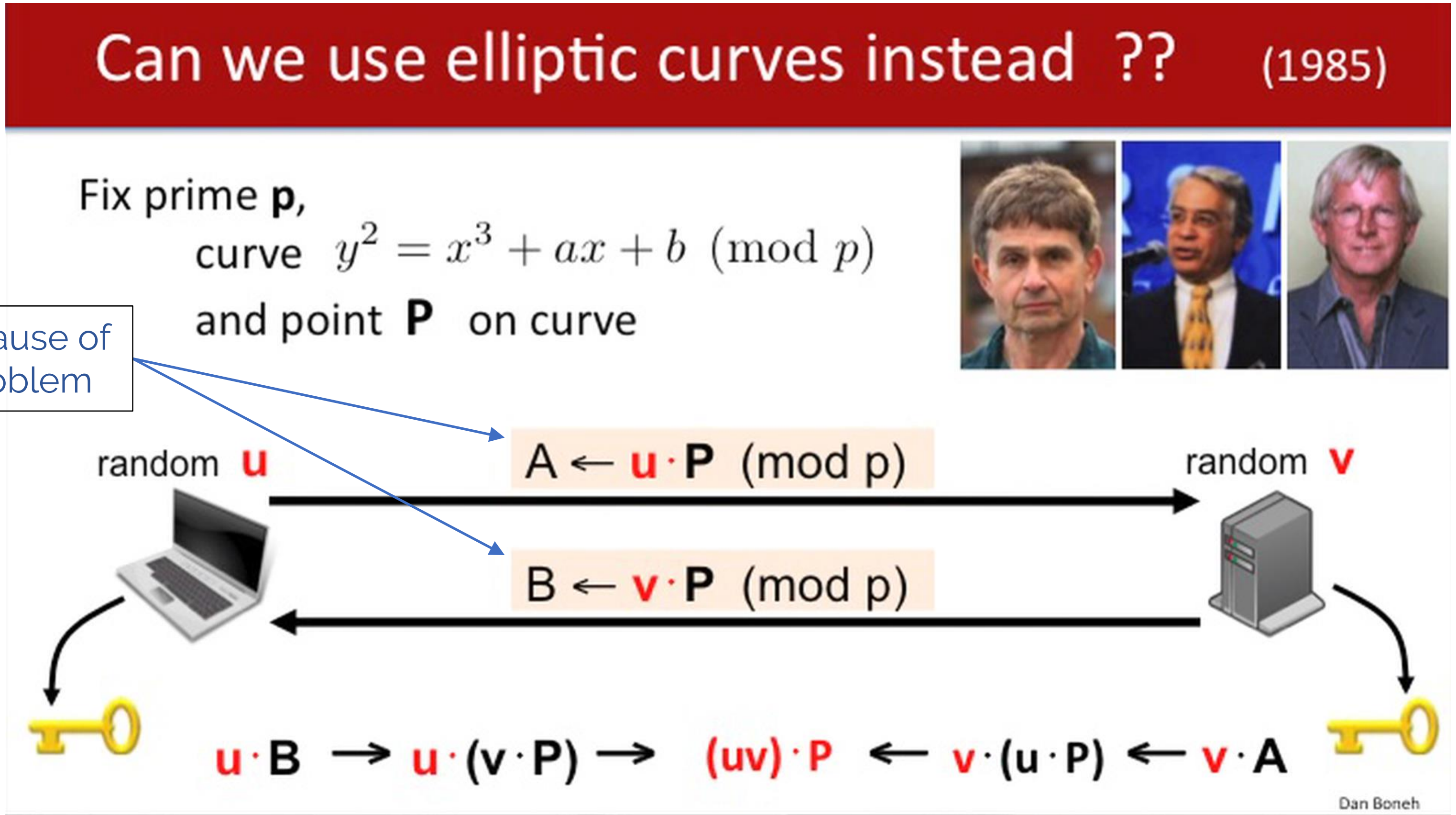


Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

One-way function because of discrete logarithm problem



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

D.1.2.3 Curve P-256

$$E: y^2 \equiv x^3 - 3x + b \pmod{p}$$

$p =$ 115792089210356248762697446949407573530086143415290314195533631308867097853951

$n =$ 115792089210356248762697446949407573529996955224135760342422259061068512044369

$b =$ 5ac635d8 aa3a93e7 b3ebbd55 769886bc 651d06b0 cc53b0f6 3bce3c3e 27d2604b

$G_x =$ 6b17d1f2 e12c4247 f8bce6e5 63a440f2 77037d81 2deb33a0 f4a13945 d898c296

$G_y =$ 4fe342e2 fe1a7f9b 8ee7eb4a 7c0f9e16 2bce3357 6b315ece cbb64068 37bf51f5

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Elliptic curves cryptography basics

Symmetric Key Size (bits)	RSA and Diffie-Hellman Key Size (bits)	Elliptic Curve Key Size (bits)
80	1024	160
112	2048	224
128	**3072**	**256**
192	7680	384
256	15360	521

End Of Part One

Questions

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Part One

- One-way Functions (Hash Functions)
- Diffie–Hellman Protocol
- RSA Protocol
- DSA Protocol
- Elliptic curves cryptography basics

Part Two

- Schnorr signature algorithm
- ECDSA protocol
- BLS protocol
- NODR crypto-protocol (BLS-based)
- Beyond modern cryptography

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Alice: secret key α $\rightarrow$ public key $A = \alpha * G$

Bob: secret key β $\rightarrow$ public key $B = \beta * G$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \bmod p$

Public parameters: a, b, p, G

Alice: secret key α $\rightarrow$ public key $A = \alpha * G$

Bob: secret key β $\rightarrow$ public key $B = \beta * G$

One-way function because of discrete logarithm problem

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Alice: secret key α $\rightarrow$ public key $A = \alpha * G$

Bob: secret key β $\rightarrow$ public key $B = \beta * G$

Signing: $s = k + \alpha * hash(m, R)$ ($R = k * G$ - random point)

Signature: (s, R)

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Alice: secret key α $\rightarrow$ public key $A = \alpha * G$

Bob: secret key β $\rightarrow$ public key $B = \beta * G$

Signing: $s = k + \alpha * hash(m, R)$ ($R = k * G$ - random point)

Signature: (s, R)

Verify: $s * G$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Alice: secret key $\alpha \rightarrow$ public key $A = \alpha * G$

Bob: secret key $\beta \rightarrow$ public key $B = \beta * G$

Signing: $s = k + \alpha * hash(m, R)$ ($R = k * G$ - random point)

Signature: (s, R)

Verify: $s * G = k * G + \alpha * hash(m, R) * G$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

Elliptic curve: $y^2 = x^3 + ax + b \ mod \ p$

Public parameters: a, b, p, G

Alice: secret key α $\rightarrow$ public key $A = \alpha * G$

Bob: secret key β $\rightarrow$ public key $B = \beta * G$

Signing: $s = k + \alpha * hash(m, R)$ ($R = k * G$ - random point)

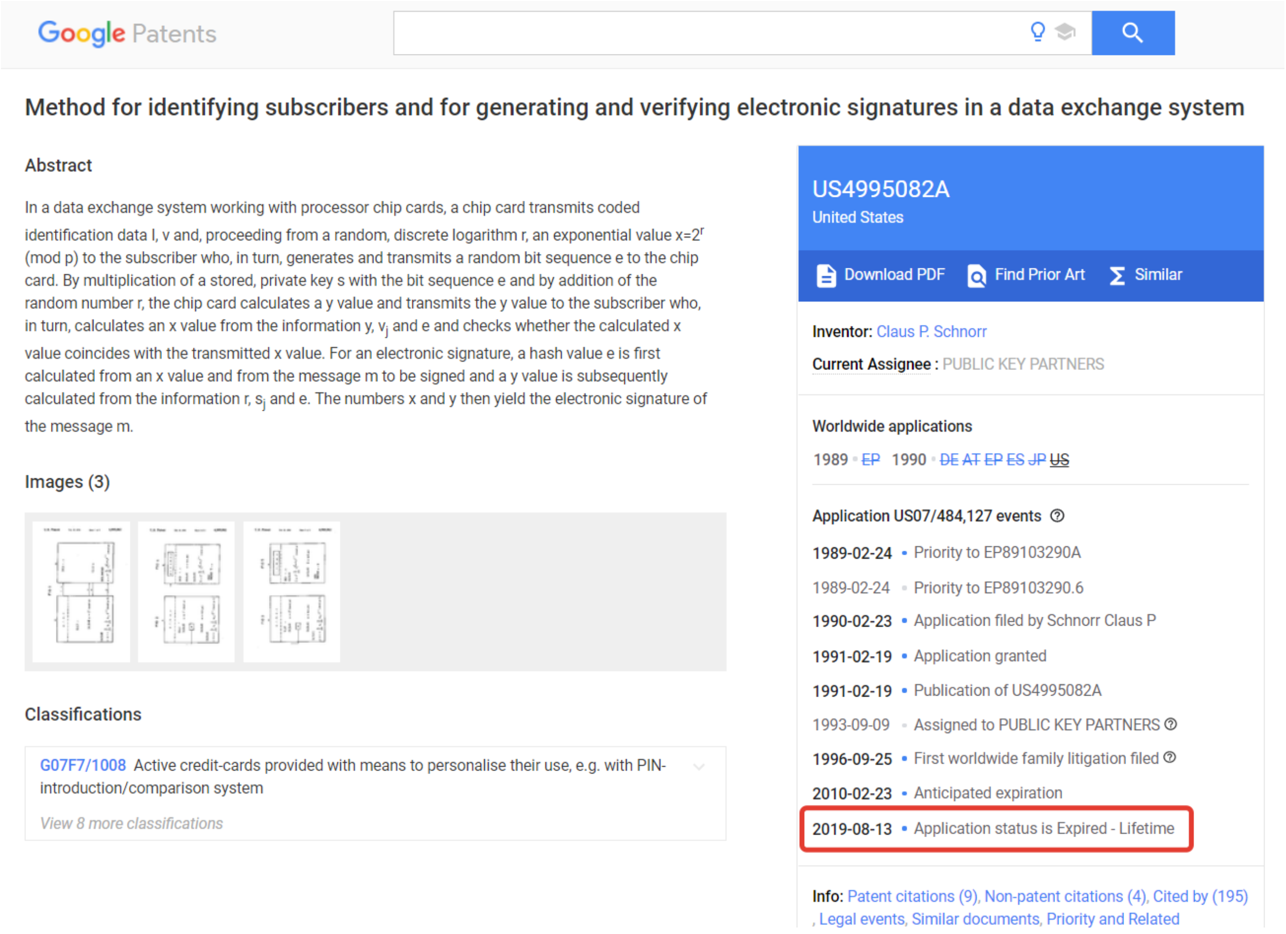
Signature: (s, R)

Verify: $s * G = R + A * hash(m, R)$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)



Google Patents

Method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system

Abstract

In a data exchange system working with processor chip cards, a chip card transmits coded identification data I, v and, proceeding from a random, discrete logarithm r, an exponential value $x=2^r$ (mod p) to the subscriber who, in turn, generates and transmits a random bit sequence e to the chip card. By multiplication of a stored, private key s with the bit sequence e and by addition of the random number r, the chip card calculates a y value and transmits the y value to the subscriber who, in turn, calculates an x value from the information y, v_j and e and checks whether the calculated x value coincides with the transmitted x value. For an electronic signature, a hash value e is first calculated from an x value and from the message m to be signed and a y value is subsequently calculated from the information r, s_j and e. The numbers x and y then yield the electronic signature of the message m.

Images (3)

Classifications

G07F7/1008 Active credit-cards provided with means to personalise their use, e.g. with PIN-introduction/comparison system

View 8 more classifications

US4995082A
United States

Download PDF | Find Prior Art | Similar

Inventor: Claus P. Schnorr
Current Assignee : PUBLIC KEY PARTNERS

Worldwide applications
1989 • EP 1990 • DE AT EP ES JP US

Application US07/484,127 events

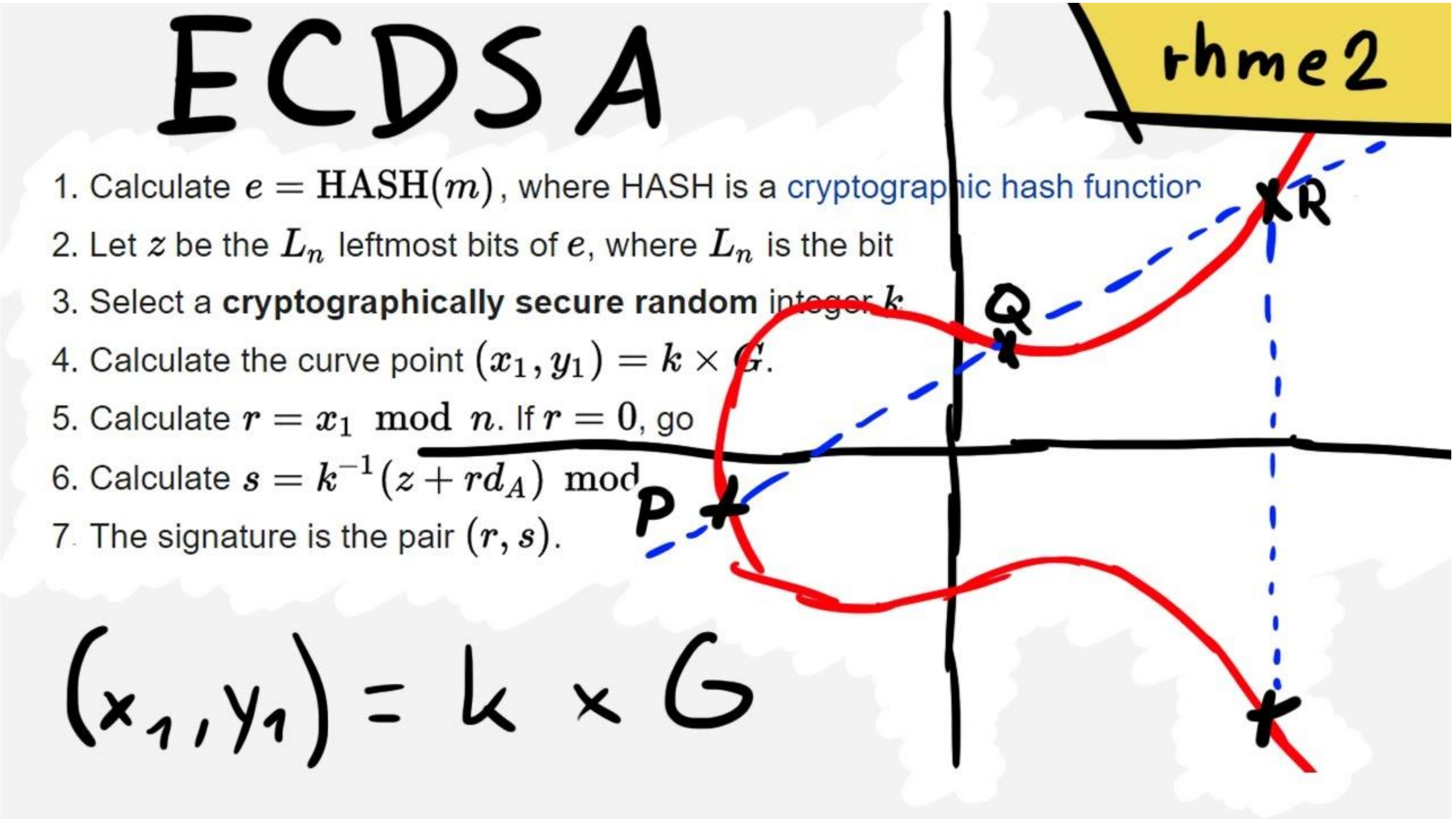
- **1989-02-24** • Priority to EP89103290A
- 1989-02-24 • Priority to EP89103290.6
- **1990-02-23** • Application filed by Schnorr Claus P
- **1991-02-19** • Application granted
- **1991-02-19** • Publication of US4995082A
- 1993-09-09 • Assigned to PUBLIC KEY PARTNERS
- **1996-09-25** • First worldwide family litigation filed
- **2010-02-23** • Anticipated expiration
- **2019-08-13** • Application status is Expired - Lifetime

Info: Patent citations (9), Non-patent citations (4), Cited by (195), Legal events, Similar documents, Priority and Related

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

ECDSA protocol (1999 ANSI, 2000 IEEE and NIST)



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol

What if CDH were easy?

Suppose poly-time alg. for CDH: $P,\ u \cdot P,\ v \cdot P \ \Rightarrow\ (uv) \cdot P$

but no poly-time algorithm for: $P,\ u \cdot P \ \not\Rightarrow\ u$ (discrete log)

Bad for key exchange ... but great for crypto !

Why? "encrypt" $m \in F_p$ as $\boxed{E(m) = m \cdot P}$

Then: $m_1 \cdot P,\ m_2 \cdot P \ \Rightarrow\ (m_1+m_2) \cdot P\ ,\ (m_1 \cdot m_2) \cdot P$

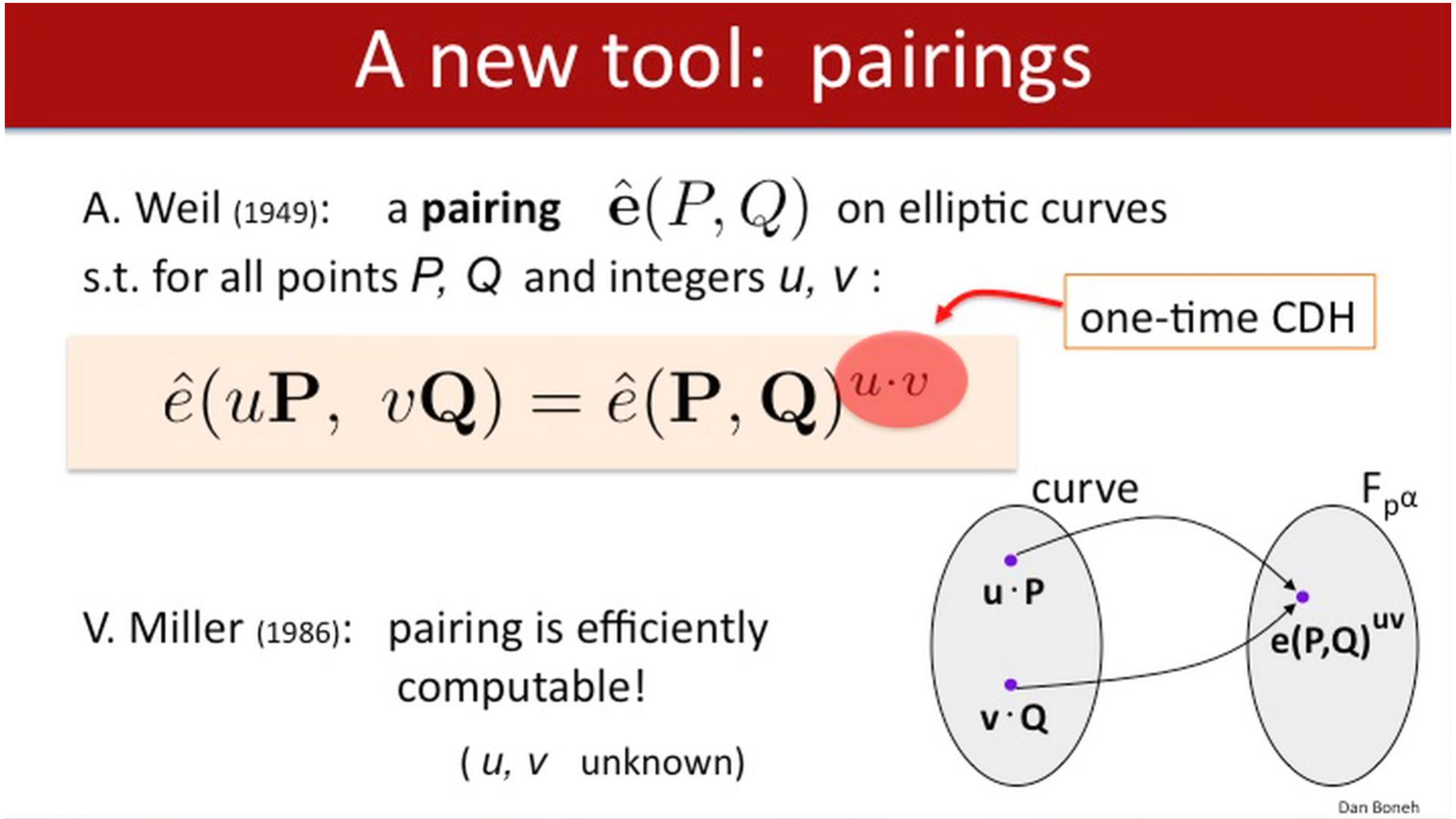
Computing on ciphertexts !! (can be made decryptable and secure)

Dan Boneh

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

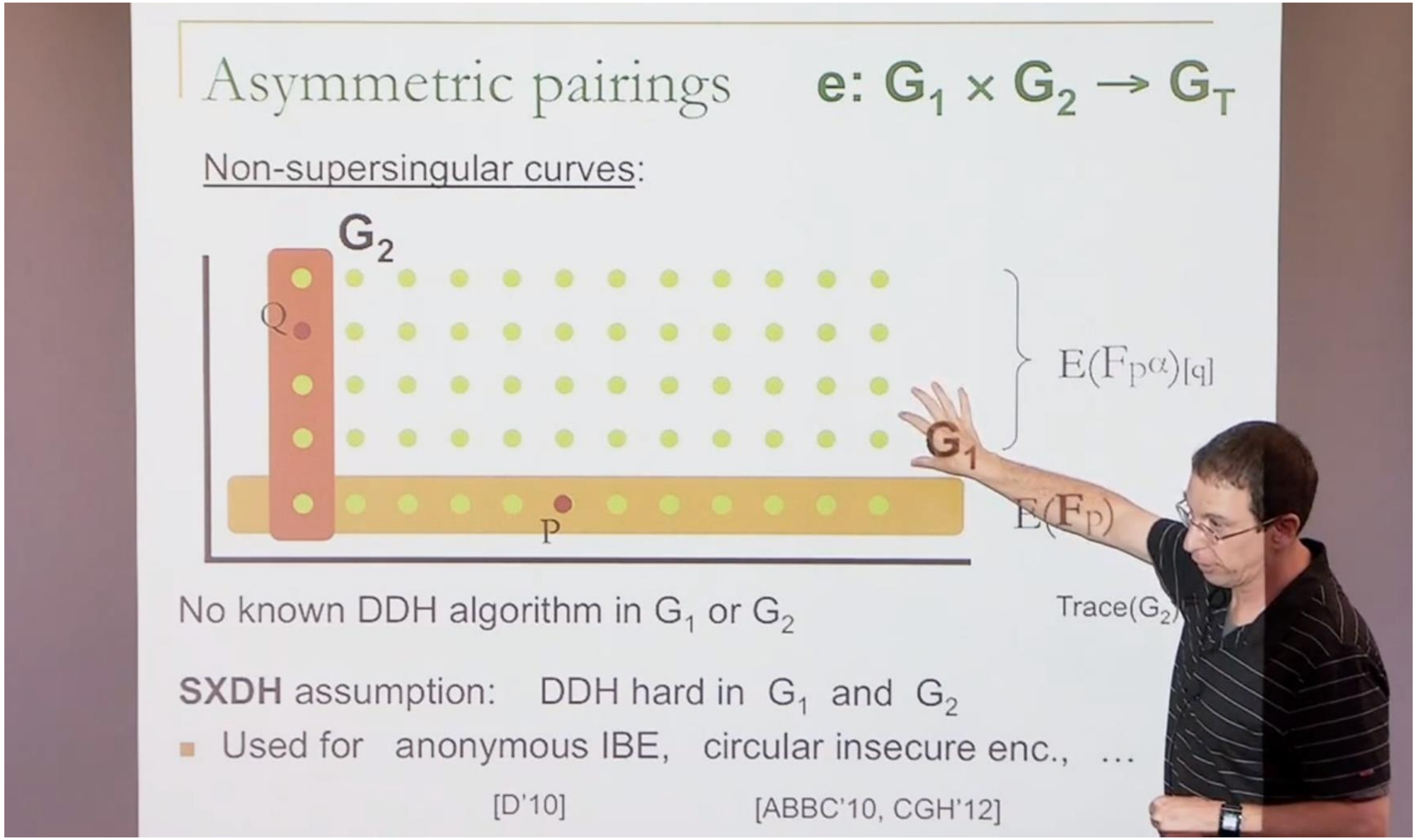
BLS protocol



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol (Boneh–Lynn–Shacham, 2004)

Elliptic curve 1: $y^2 = x^3 + a_1 x + b_1 \ mod \ p_1, \quad G_1$ (for public and private keys)

Elliptic curve 2: $y^2 = x^3 + a_2 x + b_2 \ mod \ p_2, \quad G_2$ (for hasing and signatures)

Pairing function: $e(\alpha * P, Q) = e(P, Q)^{\alpha} = e(P, \alpha * Q)$

$$e(\alpha * P, \beta * Q) = e(P, Q)^{\alpha\beta} = e(\beta * P, \alpha * Q)$$

Alice: secret key $\alpha \rightarrow$ public key $A = \alpha * G_1$

Signing: $S = \alpha * H(m)$ (no random points!)

Signature: S

Verify: $e(A, H(m)) = e(G_1, S)$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol (Boneh–Lynn–Shacham, 2004)

Elliptic curve 1: $y^2 = x^3 + a_1 x + b_1 \; mod \; p_1, \quad G_1$ (for public and private keys)

Elliptic curve 2: $y^2 = x^3 + a_2 x + b_2 \; mod \; p_2, \quad G_2$ (for hasing and signatures)

Pairing function: $e(\alpha * P, Q) = e(P, Q)^{\alpha} = e(P, \alpha * Q)$

$$e(\alpha * P, \beta * Q) = e(P, Q)^{\alpha\beta} = e(\beta * P, \alpha * Q)$$

Alice: secret key $\alpha \rightarrow$ public key $A = \alpha * G_1$

Signing: $S = \alpha * H(m)$ (no random points!)

Signature: S

Verify: $e(A, H(m)) = e(G_1, S)$

$$\boldsymbol{e(G_1, H(m))^{\alpha}}$$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol (Boneh–Lynn–Shacham, 2004)

Key aggregation and n-of-n multisignature

If we are using multisignature addresses, we are signing *the same transaction* with different keys. In this case we can do key aggregation like in Schnorr, where we combine all signatures and all keys to a single pair of a key and a signature. Let's take a common 3-of-3 multisig scheme (it can be done similarly for any number of signers).

A simple way to combine them is to add all the signatures and all the keys together. The result will be a signature $S=S1+S2+S3$ and a key $P=P1+P2+P3$. It's easy to see that the same verification equation still works:

$$e(G, S) = e(P, H(m))$$

$$e(G, S) = e(G, S1+S2+S3) = e(G, (pk1+pk2+pk3)\times H(m)) = e((pk1+pk2+pk3)\times G, H(m)) = e(P1+P2+P3, H(m)) = e(P, H(m))$$

Perm Summer School on Blockchain & Cryptomarkets 2019

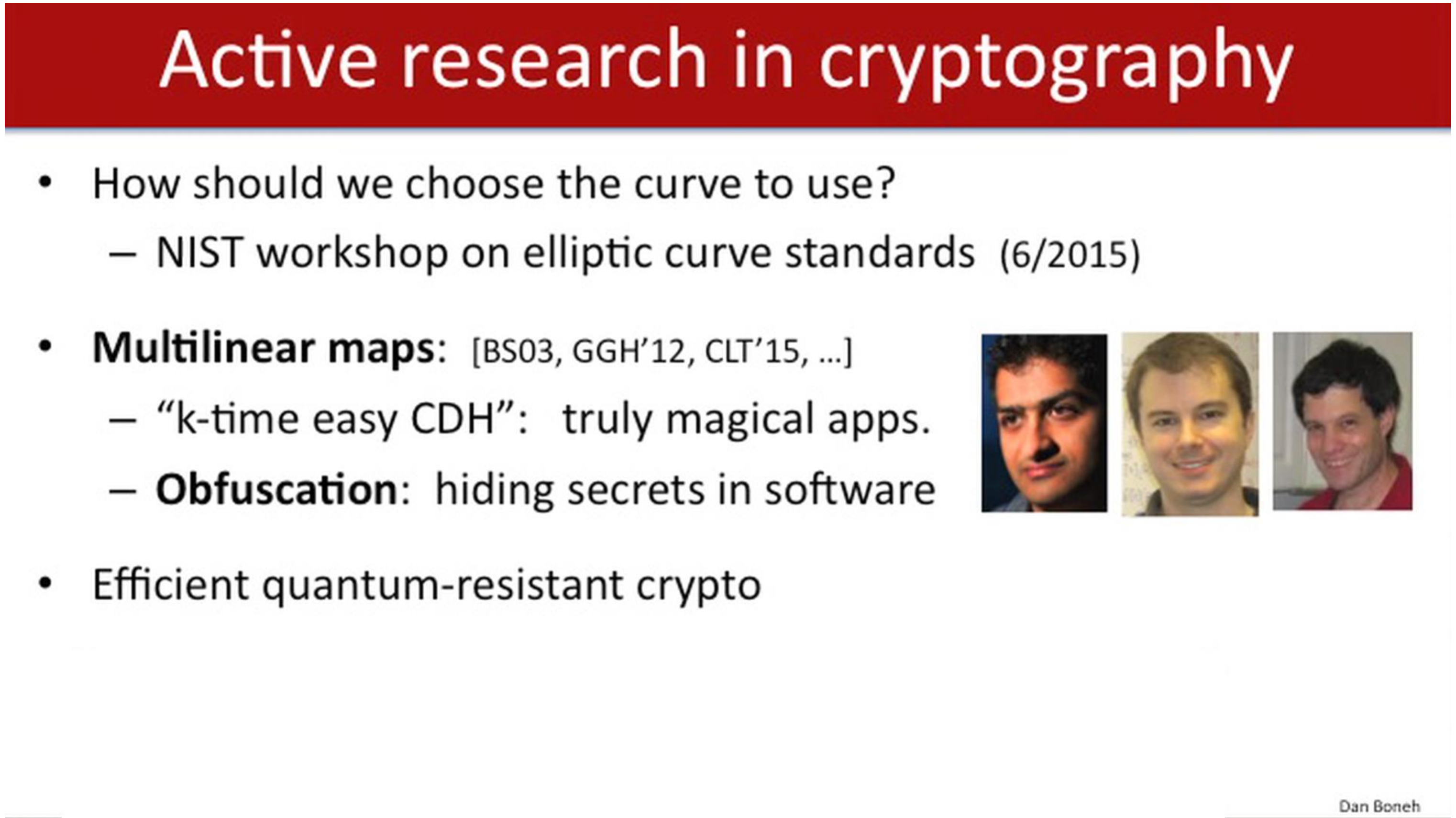
Survey on a modern cryptography workshop

NODR crypto-protocol (BLS-based)

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Beyond modern cryptography



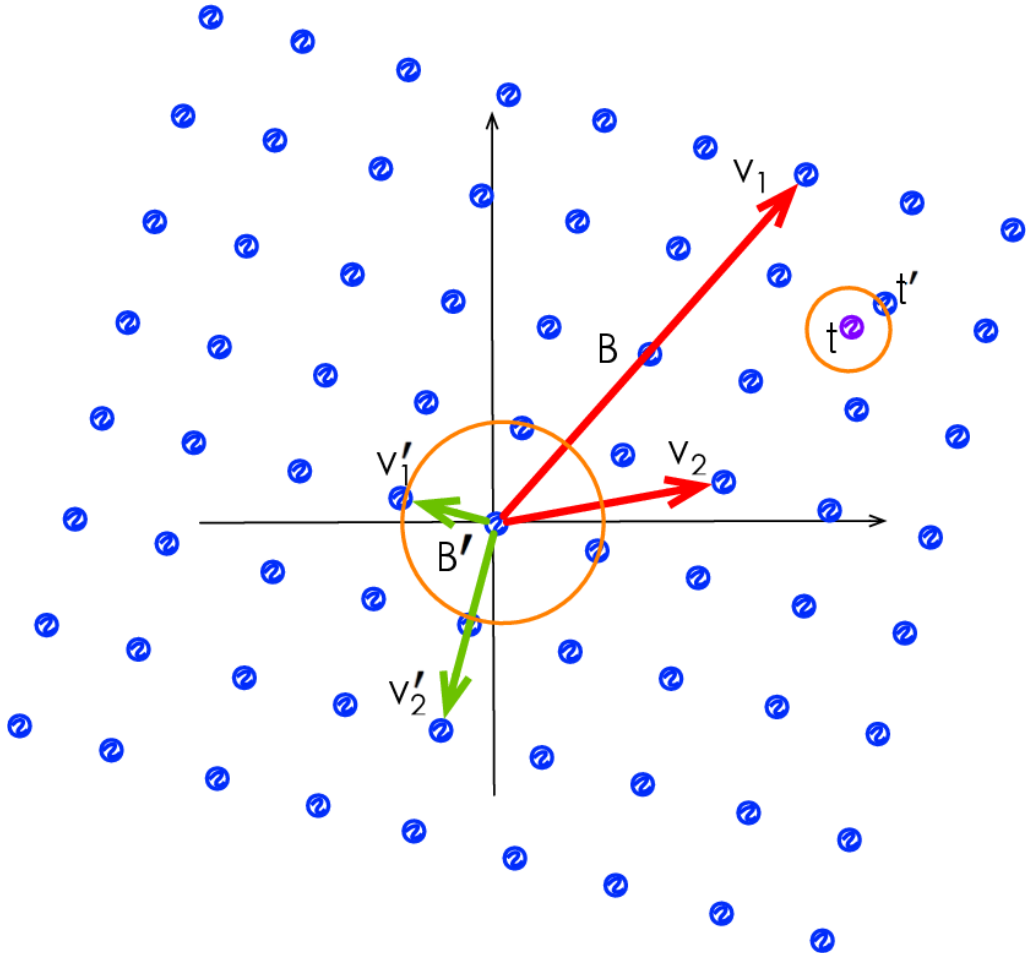
Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Beyond modern cryptography

Lattice-based cryptography

Fully homomorphic encryption



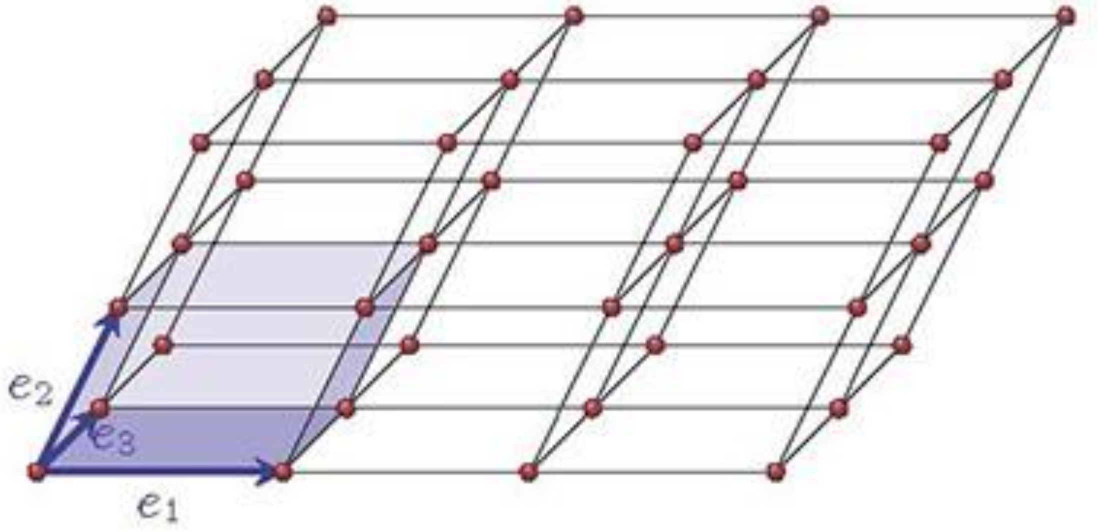
Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Beyond modern cryptography

Lattice-based cryptography

Fully homomorphic encryption



End Of Part Two

Questions