

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

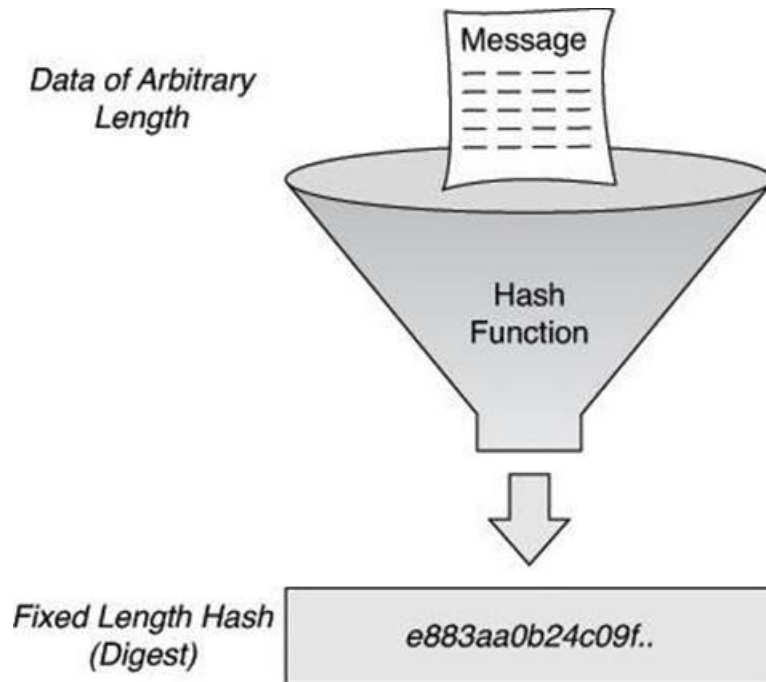
Part Two

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

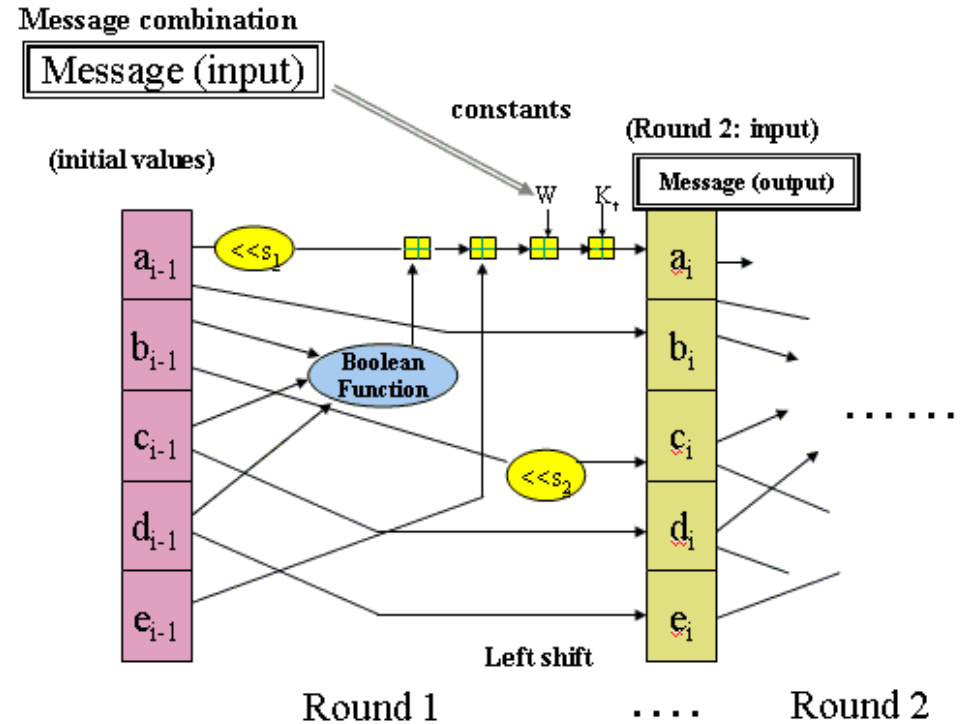
Perm Summer School on Blockchain & Cryptomarkets 2019

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

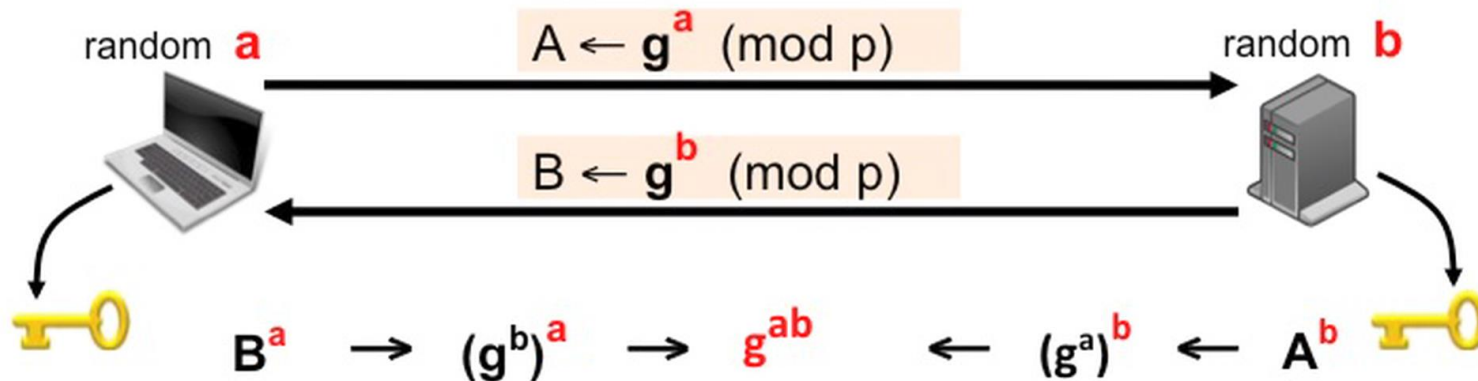
Diffie-Hellman Protocol

Diffie, Hellman, Merkle: 1976

Where do shared secret keys come from?

A remarkable solution: **(basic) Diffie-Hellman**

Fix prime p and $g \in \mathbb{F}_p$

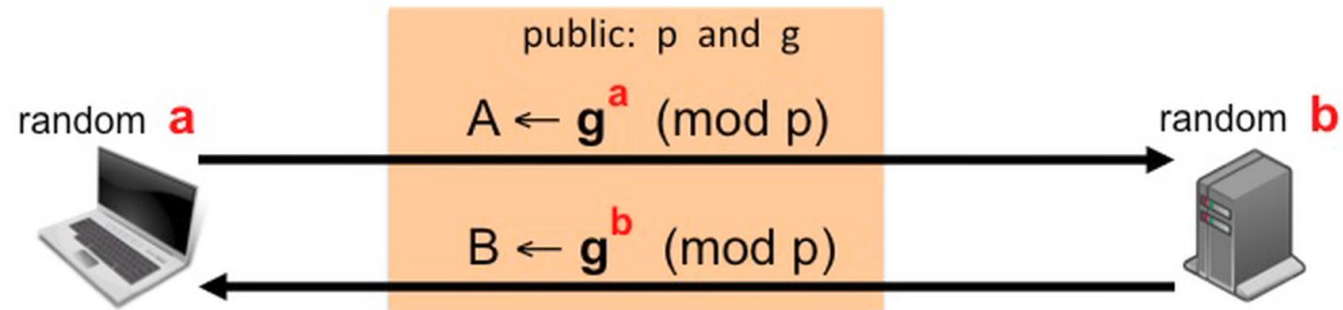


Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Diffie-Hellman Protocol

Security of Diffie-Hellman (eavesdropping only)



Eavesdropper sees: p , g , $A=g^a \pmod p$, and $B=g^b \pmod p$

Can she compute $g^{ab} \pmod p$??

CDH problem (mod p): given random (g, g^a, g^b) compute $g^{ab} \pmod p$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

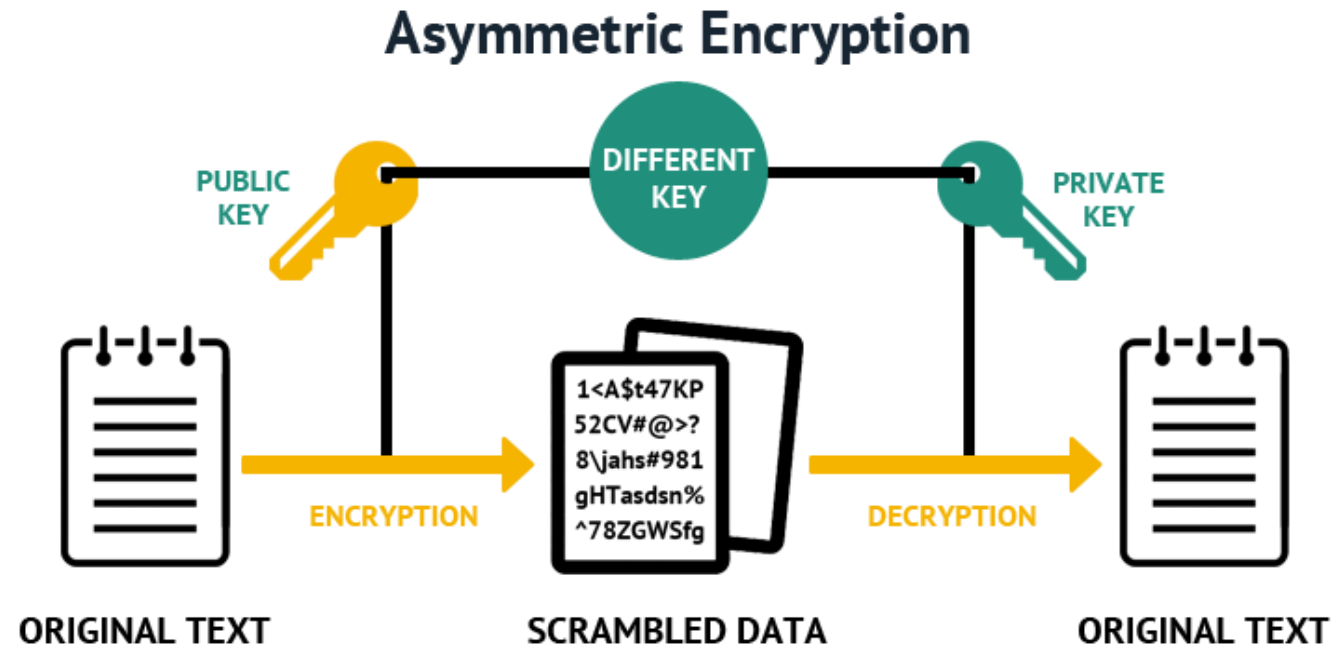
Diffie-Hellman Protocol

- 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 p and 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 $e \times d \equiv 1 \pmod{\varphi}$
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

$$c = m^e \pmod{n}$$

Decryption:

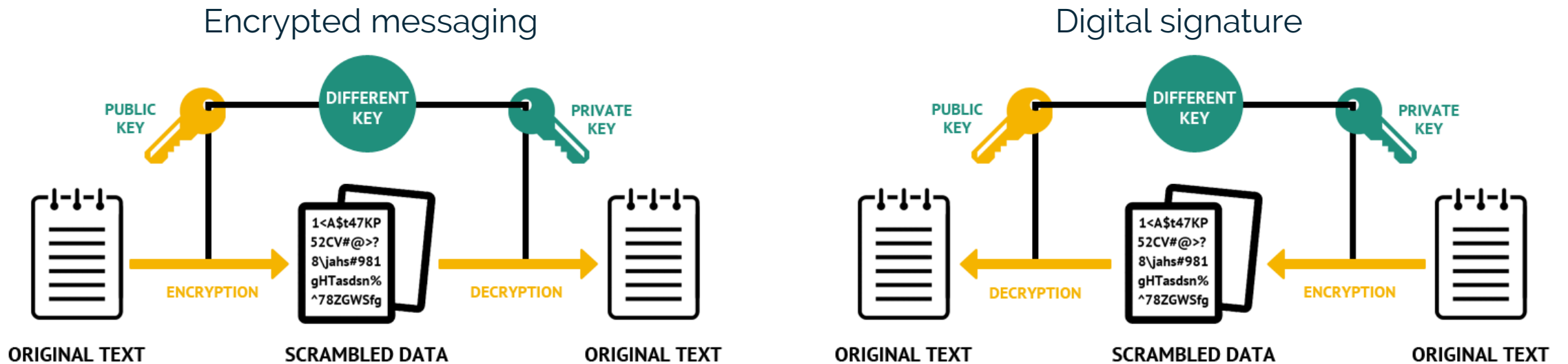
1. Decrypt the key using

$$m = c^d \pmod{n}$$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

DSA Protocol



Encrypt $ciphertext = message^e \bmod n$

Decrypt $message = ciphertext^d \bmod n$

Sing $signature = message^d \bmod n$

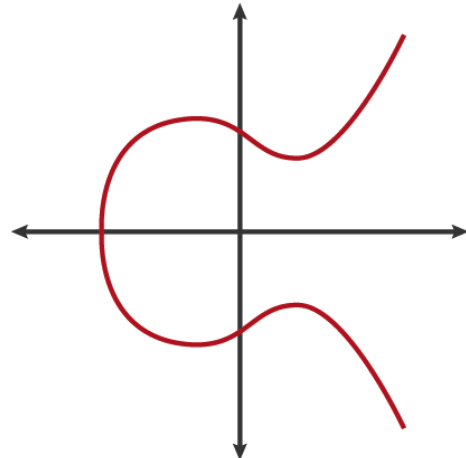
Verify $message = signature^e \bmod n$

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

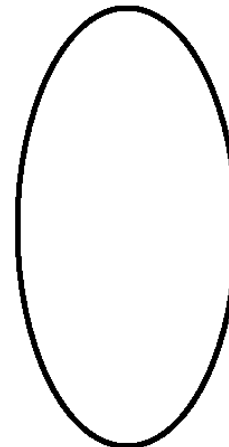
Elliptic curves cryptography basics

I DON'T UNDERSTAND WHY
PEOPLE GET CONFUSED ...
I DON'T LOOK ANYTHING
LIKE YOU!



ELLIPTIC CURVE

I THINK IT'S THE NAME!
LET'S ASK JAVA AND
JAVASCRIPT TO SEE HOW
THEY DEAL WITH IT



ELLIPSE

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

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

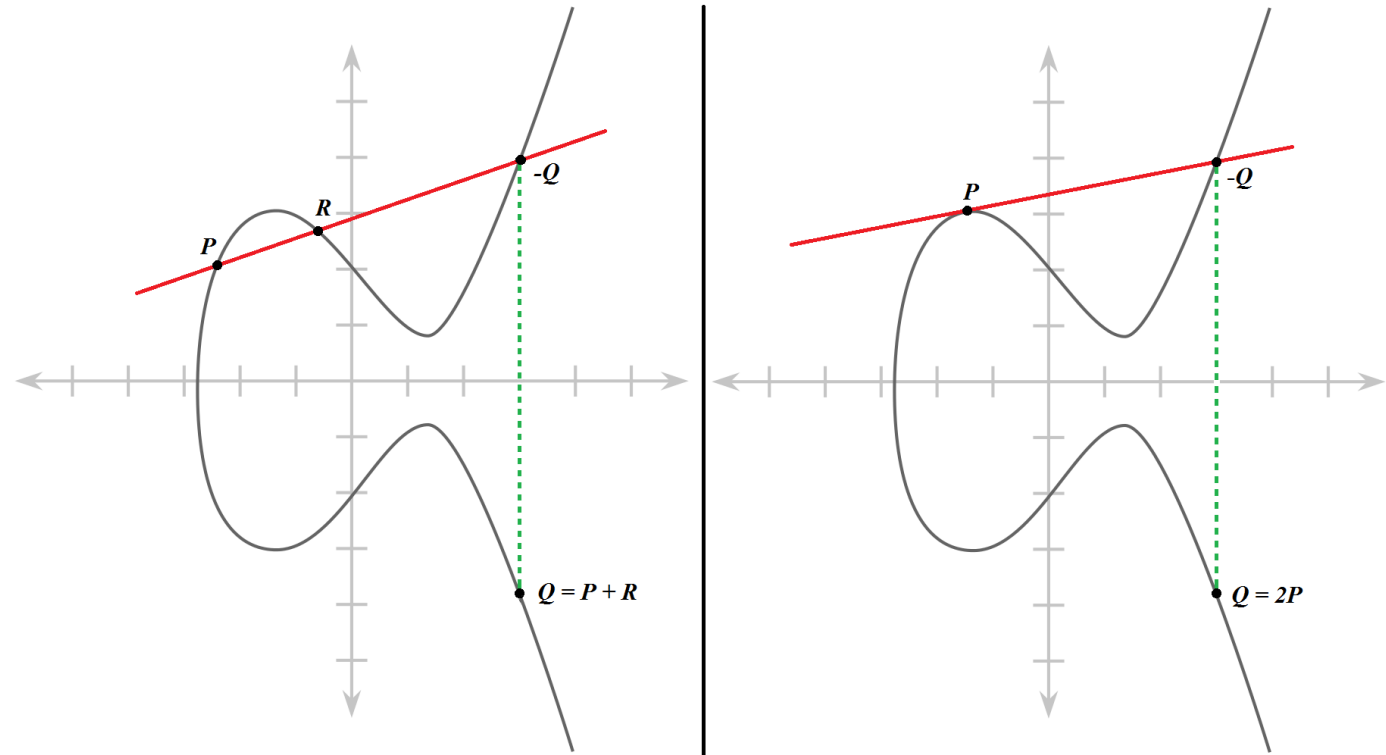
$$A + (B + C) = (A + B) + C$$

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

$$P + P = 2P$$

$$P + P + P = 3P$$

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



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:

where

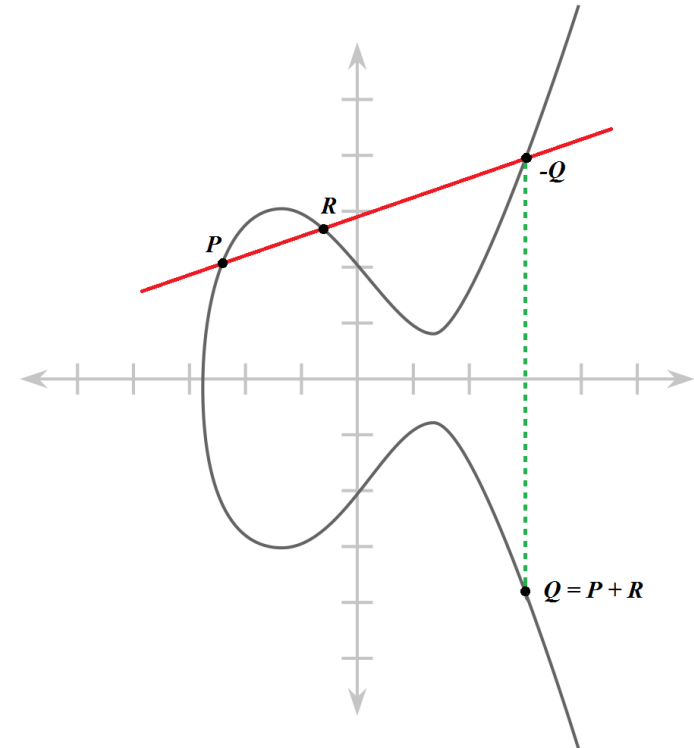
$$P_1 + P_2 = P_3 = (x_3, y_3)$$

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

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

$$y^2 = x^3 + ax + b \pmod{p}$$

Point addition operation

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

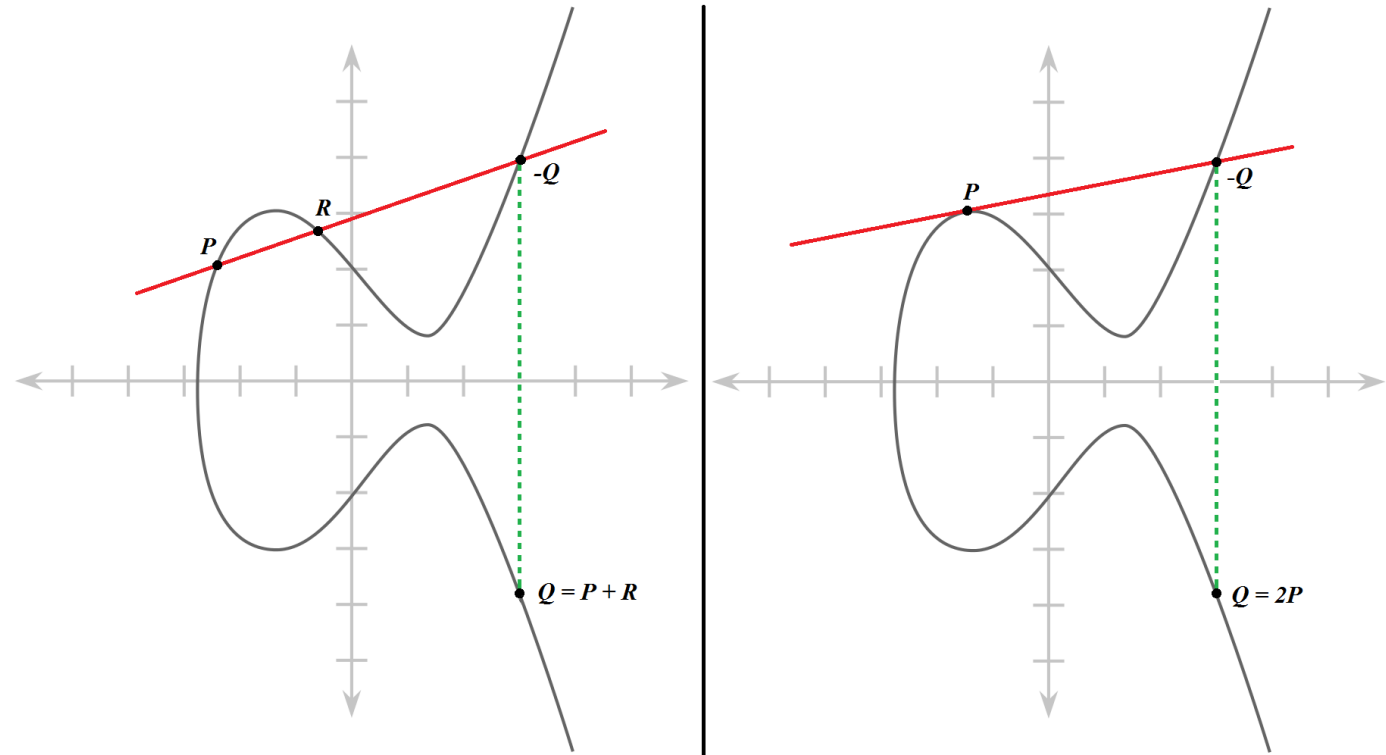
$$A + (B + C) = (A + B) + C$$

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

$$P + P = 2P$$

$$P + P + P = 3P$$

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



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: $25 \bmod 23 = 729 + 9 \bmod 23$)

The 23 points on this curve:

$(0,0)$ $(1,5)$ $(1,18)$ $(9,5)$ $(9,18)$

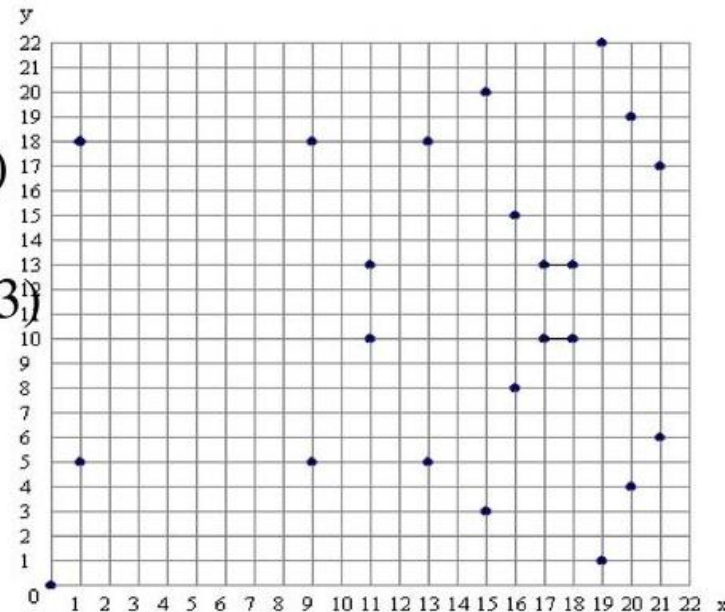
$(11,10)$ $(11,13)$ $(13,5)$ $(13,18)$

$(15,3)$ $(15,20)$ $(16,8)$ $(16,15)$

$(17,10)$ $(17,13)$ $(18,10)$ $(18,13)$

$(19,1)$ $(19,22)$ $(20,4)$ $(20,19)$

$(21,6)$ $(21,17)$



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

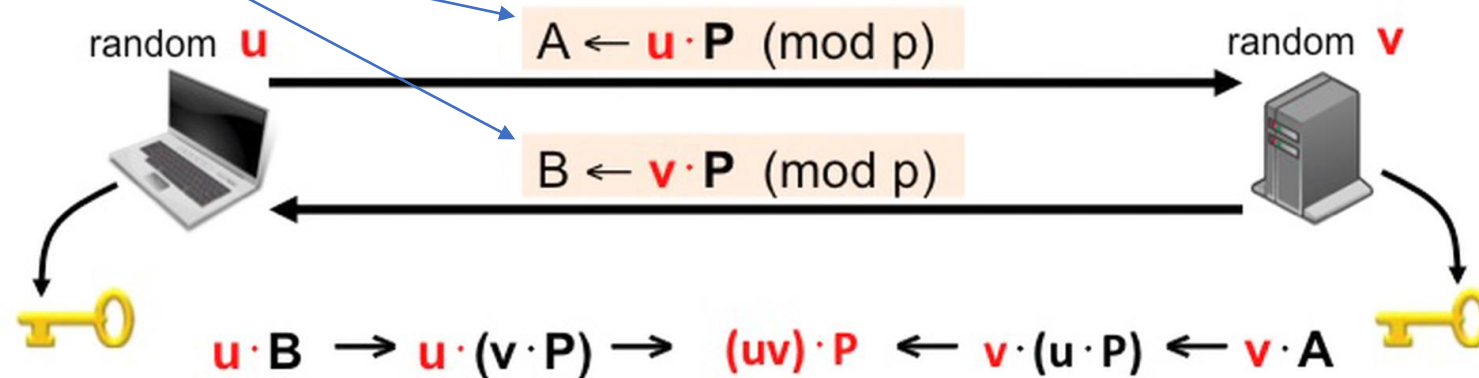
Elliptic curves cryptography basics

Can we use elliptic curves instead ?? (1985)

Fix prime p ,
curve $y^2 = x^3 + ax + b \pmod{p}$
and point P on curve



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

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

$p =$ 1157920892103562487626974469494075735300861434152903141955
33631308867097853951

$n =$ 1157920892103562487626974469494075735299969552241357603424
22259061068512044369

$b =$ 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

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

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: $y^2 = x^3 + ax + b \pmod{p}$

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: $y^2 = x^3 + ax + b \text{ mod } p$

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: $y^2 = x^3 + ax + b \text{ mod } p$

Public parameters: a, b, p, G

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

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

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: $y^2 = x^3 + ax + b \text{ mod } p$

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

Signature: (s, R)

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

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

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: $s = k + \alpha * \text{hash}(m, R)$ ($R = k * G$ - 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: $y^2 = x^3 + ax + b \text{ mod } p$

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

Signature: (s, R)

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

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Schnorr signature algorithm (1989)

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

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

Signature: (s, R)

Verify: $s * G = R + A * \text{hash}(m, R)$

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 l , v and, proceeding from a random, discrete logarithm r , an exponential value $x=2^r \pmod{p}$ 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 , 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 , 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)

ECDSA

1. Calculate $e = \text{HASH}(m)$, where HASH is a cryptographic hash function
2. Let z be the L_n leftmost bits of e , where L_n is the bit
3. Select a **cryptographically secure random integer** k
4. Calculate the curve point $(x_1, y_1) = k \times G$.
5. Calculate $r = x_1 \bmod n$. If $r = 0$, go
6. Calculate $s = k^{-1}(z + rd_A) \bmod n$
7. The signature is the pair (r, s) .

$(x_1, y_1) = k \times G$

The diagram shows an elliptic curve on a coordinate system. A red curve is plotted. A point P is marked on the curve. A point Q is marked on the curve. A point XR is marked on the curve. A vertical dashed line passes through XR and intersects the curve at another point. A yellow box labeled $rhme2$ is positioned above the curve. Handwritten annotations include P , Q , XR , and a vertical dashed line.

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 $E(m) = m \cdot P$

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)

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol

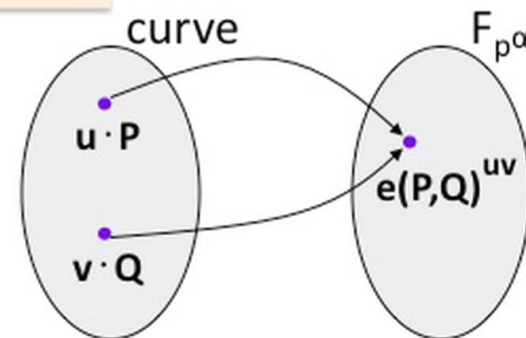
A new tool: pairings

A. Weil (1949): a **pairing** $\hat{e}(P, Q)$ on elliptic curves
s.t. for all points P, Q and integers u, v :

$$\hat{e}(u\mathbf{P}, v\mathbf{Q}) = \hat{e}(\mathbf{P}, \mathbf{Q})^{u \cdot v}$$

one-time CDH

V. Miller (1986): pairing is efficiently
computable!
(u, v unknown)



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol

Asymmetric pairings $e: G_1 \times G_2 \rightarrow G_T$

Non-supersingular curves:

No known DDH algorithm in G_1 or G_2

SXDH assumption: DDH hard in G_1 and G_2

- Used for anonymous IBE, circular insecure enc., ...

[D'10] [ABBC'10, CGH'12]

Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

BLS protocol (Boneh–Lynn–Shacham, 2004)

Elliptic curve 1: $y^2 = x^3 + a_1x + b_1 \text{ mod } p_1, G_1$ (for public and private keys)

Elliptic curve 2: $y^2 = x^3 + a_2x + b_2 \text{ mod } p_2, G_2$ (for hashing 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: $y^2 = x^3 + a_1x + b_1 \text{ mod } p_1, G_1$ (for public and private keys)

Elliptic curve 2: $y^2 = x^3 + a_2x + b_2 \text{ mod } p_2, G_2$ (for hashing 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: $e(A, H(m)) = e(G_1, S)$

$$\stackrel{\parallel}{=} e(G_1, H(m))^\alpha$$

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

Active research in cryptography

- How should we choose the curve to use?
 - NIST workshop on elliptic curve standards (6/2015)
- **Multilinear maps:** [BS03, GGH'12, CLT'15, ...]
 - “k-time easy CDH”: truly magical apps.
 - **Obfuscation:** hiding secrets in software
- Efficient quantum-resistant crypto



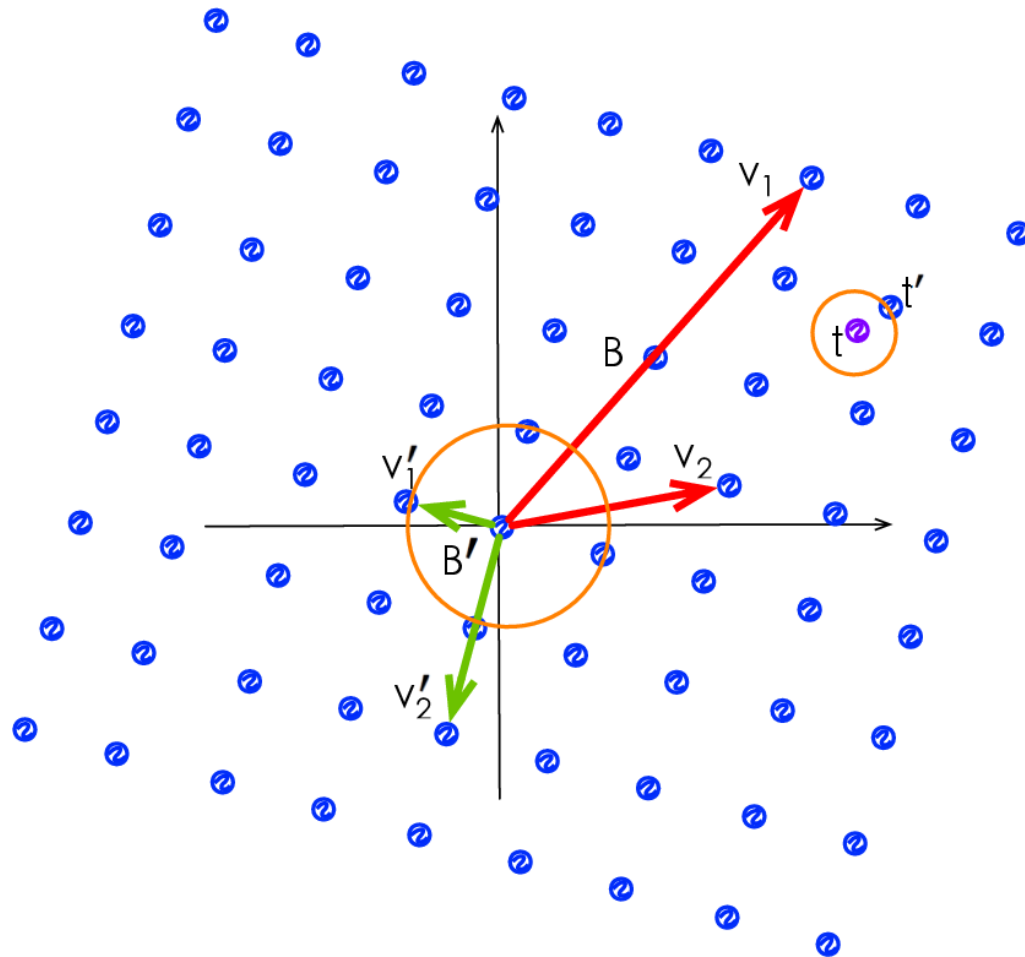
Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Beyond modern cryptography

Lattice-based cryptography

Fully homomorphic encryption



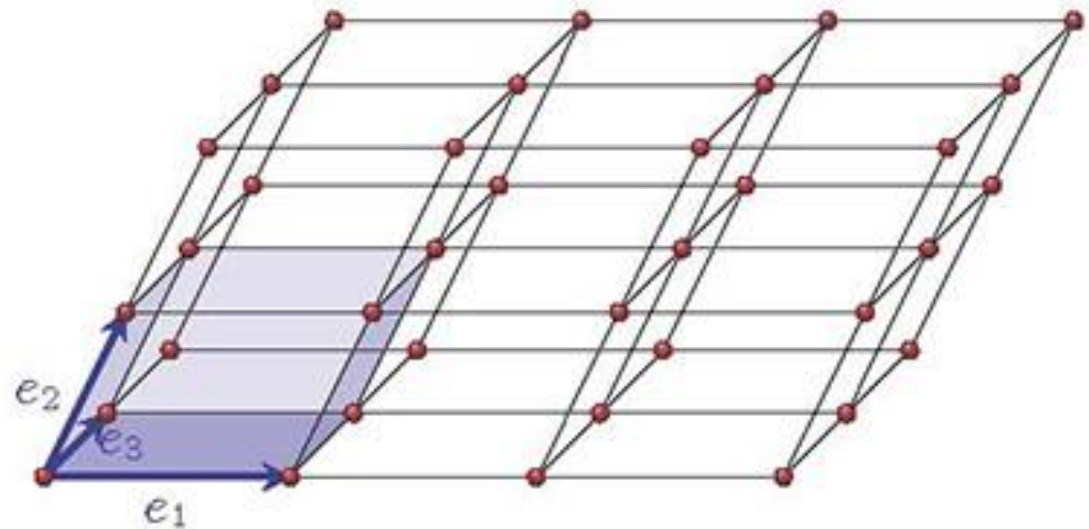
Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

Beyond modern cryptography

Lattice-based cryptography

Fully homomorphic encryption



Perm Summer School on Blockchain & Cryptomarkets 2019

Survey on a modern cryptography workshop

End Of Part Two

Questions