DL CRYPTOGRAPHY

PHONG NGUYEN

http://www.di.ens.fr/~pnguyen

DL VS RSA

- DL provides the same functionalities as RSA: encryption and signature.
- DL is more efficient than RSA.
- But DL requires more advanced mathematics.

REMEMBER DIFFIE-HELLMAN

PUBLIC-KEY CRYPTOGRAPHY

- Invented by Diffie and Hellman in 1976.

- « We stand today on the brink of a revolution in cryptography. »
- « however, it 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 a private courier or registered mail. (…) The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks. »

DISCRETE LOGARITHM

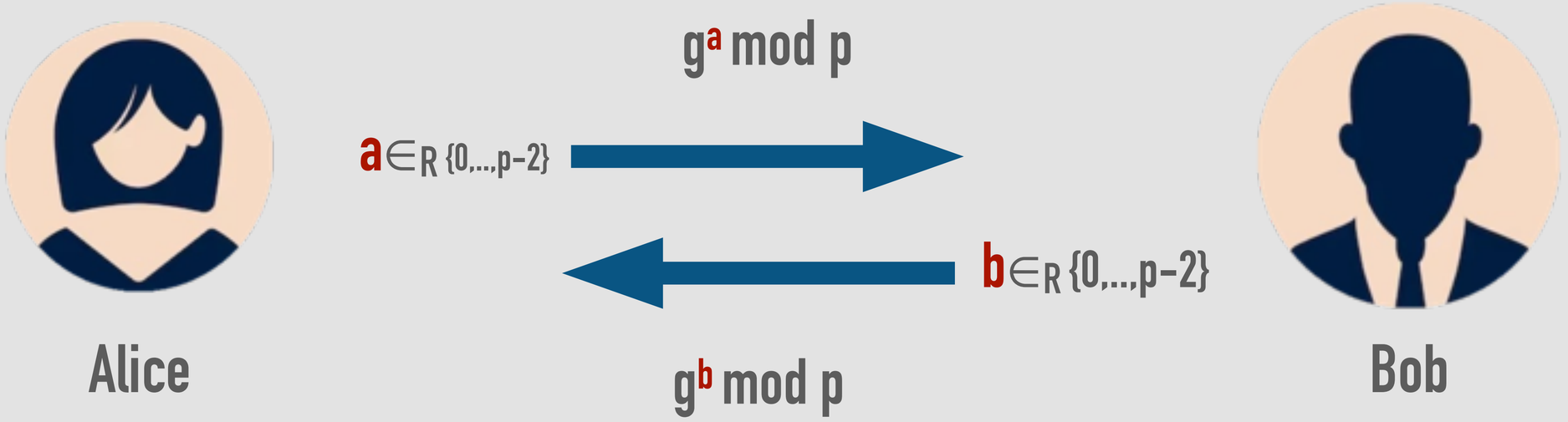
- This is the first problem used by public-key cryptography.
- In its simplest form, one selects a large prime number p, say a 2048-bit prime number.
- Let G={1,2,…,p-1}.
- Th: If p is a prime number, there exists an integer g ∈G such that for any y∈G,

  there exists a unique **x**∈{1,2,…,p-1} such that $y=g^x \bmod p$.

  Such a g can be found efficiently, given p.

  Finding **x** is the Discrete Logarithm Problem (DL).

DIFFIE-HELLMAN KEY EXCHANGE

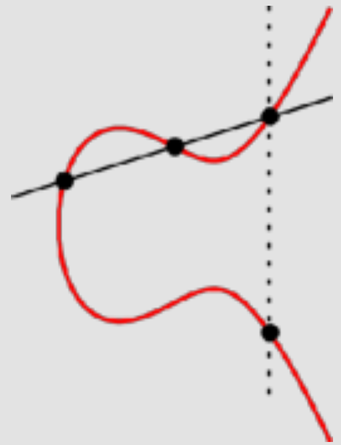


- Both can compute the shared key $g^{ab} \bmod p$.
- This key exchange is the core of El Gamal public-key encryption.

GENERALIZED DIFFIE-HELLMAN

DISCRETE LOGARITHM

- The main alternative to RSA is public-key cryptography based on the discrete logarithm problem, especially with elliptic curves: elliptic curve crypto is called ECC, proposed by Koblitz and Miller in the mid-80's.
    - 90% of Internet certificates rely on RSA signatures: the other 10% use ECDSA signatures based on the hardness of discrete logarithm over elliptic curves.



ELLIPTIC CURVE CERTIFICATE

AffirmTrust Premium ECC
Autorité de certification racine
Expire le lundi 31 décembre 2040 15:20:24 heure normale d'Europe centrale
Ce certificat est valide

▸ **Se fier**
▾ **Détails**

Sujet	
Pays	US
Organisation	AffirmTrust
Nom	AffirmTrust Premium ECC
Nom de l'émetteur	
Pays	US
Organisation	AffirmTrust
Nom	AffirmTrust Premium ECC
Numéro de série	8401224907861490260
Version	3
Algorithme de signature	Signature ECDSA avec SHA-384 (1.2.840.10045.4.3.3)
Paramètres	aucun
Non valide avant	vendredi 29 janvier 2010 15:20:24 heure normale d'Europe centrale
Non valide après	lundi 31 décembre 2040 15:20:24 heure normale d'Europe centrale
Infos de clé publique	
Algorithme	Clé publique à courbe elliptique (1.2.840.10045.2.1)
Paramètres	Courbe elliptique secp384r1 (1.3.132.0.34)
Clé publique	97 octets : 04 0D 30 5E 1B 15 9D 03 …
Dimension de clé	384 bits
Utilisation de la clé	Vérifier
Signature	102 octets : 30 64 02 30 17 09 F3 87 …

GROUPS

- A set G with a binary law $\odot$ is a group when:
  - **Internal law**: $a \odot b \in G$ for all $a,b \in G$
  - **Associativity**: $(a \odot b) \odot c = a \odot (b \odot c)$ for any $a,b,c \in G$
  - **Neutral element**: there exists $e \in G$ such that for any $a \in G$, $a \odot e = e \odot a = a$.
  - **Inversion**: for any $a \in G$, there exists $b \in G$ such that $a \odot b = b \odot a = e$. We write: $b = a^{-1}$.

EXAMPLES OF GROUPS

- {0,1} with which law ?
- The integers modulo n, with modular addition.
- The integers $\mathbf{Z}$, with addition.
- The non-zero rationals $\mathbf{Q}^*$, with multiplication.

ORDERS

- Let $(G,\odot)$ be a group. We define the notation g^n for $g \in G$ and $n \in \mathbf{Z}$:
  $g^0=e$, $g^1=g$, $g^2=g\odot g$, g^{-1} is the inverse of g. Then $g^{a+b} = g^a \odot g^b$ holds for any $a,b \in \mathbf{Z}$.

- If there exists $n>0$ such that $g^n=e$, we say that g has **finite order**: we call the **order** of g the smallest such $n>0$.

- If n is the order of g, then $g^a=g^b$ if and only if $g^{a-b}=e$ that is $a=b \bmod n$.
  Because $g^{a-b}=e$ if and only if $a-b$ is divisible by n..

FINITE FIELDS

- For any prime power $q=p^n$, there exists a “unique” finite field with exactly q elements, called GF(q).
  - Two operations + and x with neutral elements 0 and 1.
  - Every element y has an opposite -y.
  - Every nonzero element y has an inverse 1/y.

- AES uses the finite field $GF(256)=GF(2^8)$.

FINITE FIELDS

- Addition is cheap
- Multiplication is much more expensive than addition.
- Inversion is asymptotically as fast as multiplication, but in practice, there is a significant constant, so it’s better to minimize the number of inversions.

DL GROUPS FOR CRYPTOGRAPHY

- The non-zero integers mod p, where p is a prime number.

- More generally, the multiplicative group of any finite field: all the field elements, except zero.

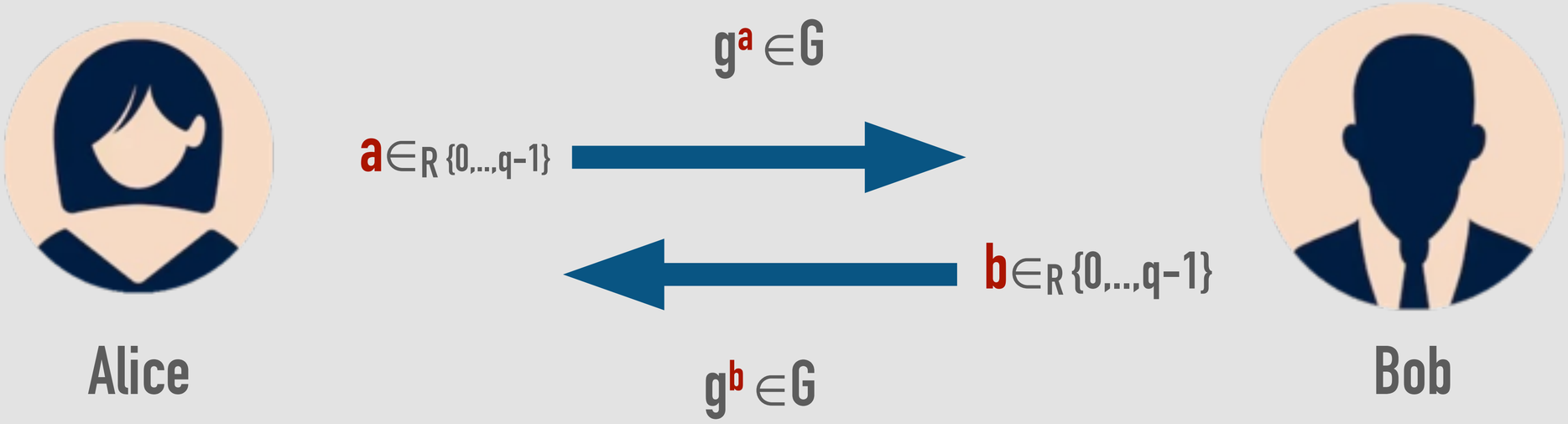
- An elliptic curve over a finite field: these groups provide the hardest DL known.

THE DISCRETE LOGARITHM PROBLEM

- Consider a cyclic group G generated by g of prime order q: $G=\{1,g,g^2,\ldots,g^{q-1}\}$.
- Any element y of G can be written as $y=g^x$ for some $0 \leq x < q$.
- The Discrete Logarithm Problem asks to find this index x for a randomly chosen y in G.
- Using well-chosen elliptic curves over finite fields, the best algorithm known is exponential in the size of the cyclic group: it requires $\sqrt{q}$ operations.
- By comparison, the best factoring algorithm is subexponential.

DIFFIE-HELLMAN KEY EXCHANGE (1976)

- Let $G=\langle g \rangle$ be generated by g of order q.



- Both can compute the shared key $g^{ab}=(g^a)^b=(g^b)^a$
- This key exchange is the core of El Gamal public-key encryption.

EL GAMAL ENCRYPTION (1984)

- Let G be a cyclic group $\langle g \rangle$ of order q.
- Secret key $x \in \mathbf{Z}_q$. Public key $y = g^x \in G$.
- Encrypt $m \in G$ as $(a,b) \in G^2$.
  - $a = g^k \in G$ where $k \in \mathbf{Z}_q$
  - $b = m y^k \in G$
- Decrypt (a,b) by recovering $y^k = g^{kx} = a^x$ then $m = b(y^k)^{-1}$

EL GAMAL ENCRYPTION (1984)

- Let G be a cyclic group $\langle g \rangle$ of order q.
- Secret key $x \in \mathbf{Z}_q$. Public key $y = g^x \in G$.

- Behind El Gamal, there is the Diffie-Hellman key exchange.

➤Alice has a secret key $x \in_R \mathbf{Z}/q\mathbf{Z}$ and discloses $y = g^x \in G$

➤Bob selects a one-time key $k \in_R \mathbf{Z}/q\mathbf{Z}$ and discloses $g^k \in G$

➤Both can compute the shared key g^{kx}.

EXAMPLE

```
Chiffrement El Gamal
p =  126765060022822940150518298884 3
générateur g =  5
Alice : ma clef secrete est x = 41569289842288043598953039560 5  et
ma clef publique est y = 109680812272966460973465966760 2
Le messsage est m = 16839048005634048773195436880 7
Le chiffrement de m est : a = 48014293169997129392294297659 1
et b = 20097886642324748014961606348 6
On dechiffre en  16839048005634048773195436880 7
```

SAGE

- print("Chiffrement El Gamal")
- p = next_prime(2^100+8479783443)
- print("p = ",p)
- F = GF(p)
- g = F(primitive_root(p))
- print("générateur g = ",g)
- x = randrange(p)
- y = g^x
- print("Alice : ma clef secrete est x =",x," et ma clef publique est y =",y)
- m = randrange(p)
- print("Le messsage est m =",m)
- k = randrange(p-1)
- a = g^k
- b = m*y^k
- print("Le chiffrement de m est : a =",a," et b =",b)
- c = a^x
- print("On dechiffre en ",b/c)

EL GAMAL SIGNATURE AND DSA

EL GAMAL SIGNATURE (1984)

- Let G be a cyclic group $\langle g \rangle$ of order q.
- Secret key $x \in \mathbf{Z}_q$. Public key $y = g^x \in G$.

- To sign $H(m) \in \mathbf{Z}_q$.
  - ➤Select a random k mod q invertible mod q
  - ➤$r = (g^k) \bmod q$.
  - ➤$s = (H(m) + xr)\, k^{-1} \bmod q$
  - ➤The signature is (r,s)
- Verify that $0 < r < q$ and $0 < s < q$
  - ➤Compute $a = H(m)\, s^{-1} \bmod q$ and $b = r\, s^{-1} \bmod q$
  - ➤Check that $r = (g^a y^b) \bmod q$

DSA (1993)

- DSA is a US standard, a variant of Schnorr's signature. Both are much more efficient than El Gamal signature.

- Common parameters
  - ➤Two large prime numbers p and q such that q divides p-1 and q is much smaller than p. Originally, q was 160-bit and p was 1024-bit.
  - ➤An integer g mod p such that g has order q.

DSA (1993)

- Secret key $x \in_R \{0,1,\ldots,q-1\}$. Public key $y=g^x \bmod p$.
- To sign $H(m) \in \{0,1,\ldots,q-1\}$.
  - ➤ Select a random k mod q which is non-zero, therefore invertible mod q.
  - ➤ $r=(g^k \bmod p) \bmod q$
  - ➤ $s= (H(m)+xr)k^{-1} \bmod q$
  - ➤ The signature is (r,s)
- Verify that $0<r<q$, $0<s<q$
  - Compute $a = H(m)s^{-1} \bmod q$ and $b = rs^{-1} \bmod q$
  - Check that $r = (g^a y^b \bmod p) \bmod q$.

IN PRACTICE

- The main competitor of RSA signatures is the Elliptic Curve variant of DSA, which gives smaller keys and signatures.

- Intuitively, $(\mathbb{Z}/\mathbf{p}\mathbb{Z})^x$ is replaced by a « small » elliptic curve: instead of working with a 2048-bit prime number, we can use an elliptic curve defined over a 256-bit finite field, such as GF(p) for 256-bit prime.

PS3

- The PS3 uses ECDSA to authentify executable codes.
- ECDSA requires a pseudo-random number k at each signature generation, like most DL-based schemes.

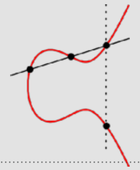
- It is well-known that one must be extremely careful with k:
  - ➤ If k is disclosed, or even a few bits...
  - ➤If k is reused...

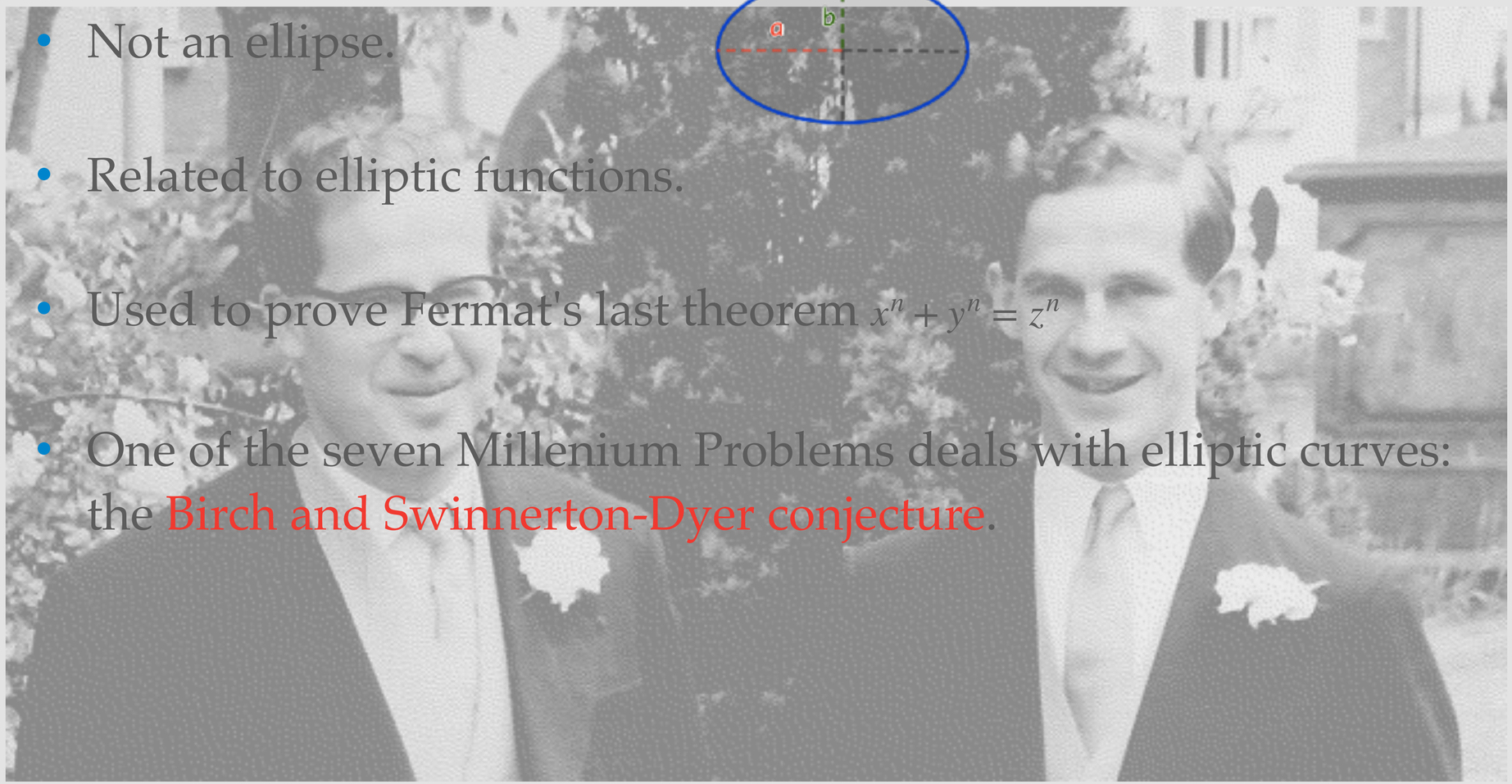
PS3'S EPIC BUG

- failOverflow announced at 27c3 in December 2010 that they were able to recover Sony's signing keys, because the same k was reused at least once, if not always...
- GeoHot put some secret keys on the web, shortly after in January 2011, but apparently not the root key.
- In response, Sony filed a lawsuit. disclosed, or even a few bits...

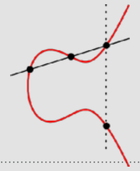
THE IRRUPTION OF ELLIPTIC CURVES

ELLIPTIC CURVES





- Not an ellipse.
- Related to elliptic functions.
- Used to prove Fermat's last theorem $x^n + y^n = z^n$
- One of the seven Millenium Problems deals with elliptic curves: the Birch and Swinnerton-Dyer conjecture.

ELLIPTIC CURVES  IN CRYPTOLOGY

- 1983: Lenstra uses elliptic curves to factor numbers.

  The ECM factors in expected subexponential time $2^{\sqrt{\log p \log\log p}}$ where p is the smallest prime factor.

- 1985: Koblitz and Miller independently proposed to use elliptic curves over finite fields for discrete logarithm cryptography.

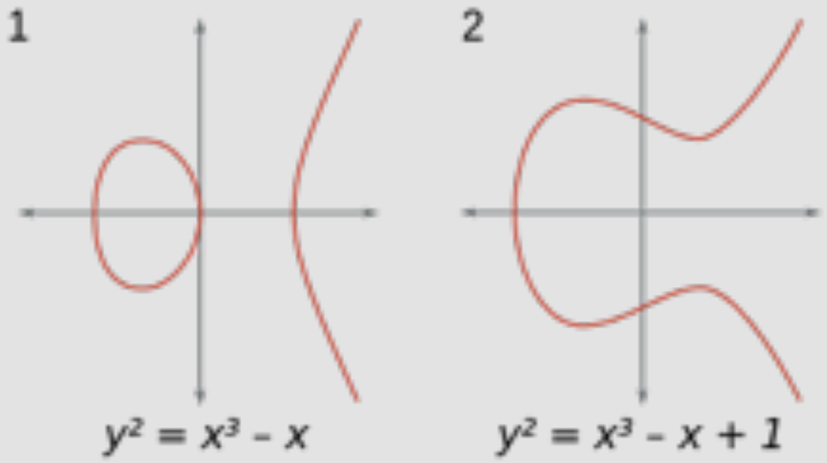
  For well-chosen elliptic curves of order q, the best DL algorithm costs $\sqrt{q}$.

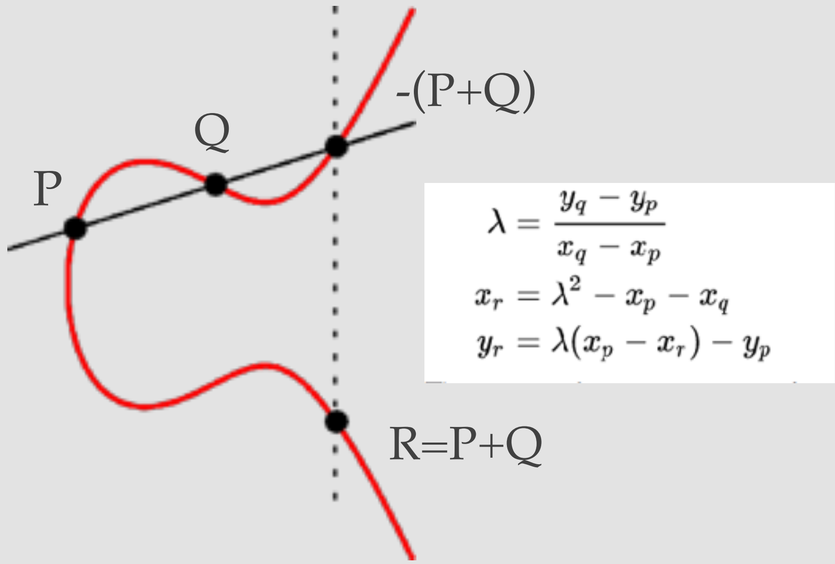
Neal Koblitz

Victor Miller

ELLIPTIC CURVES

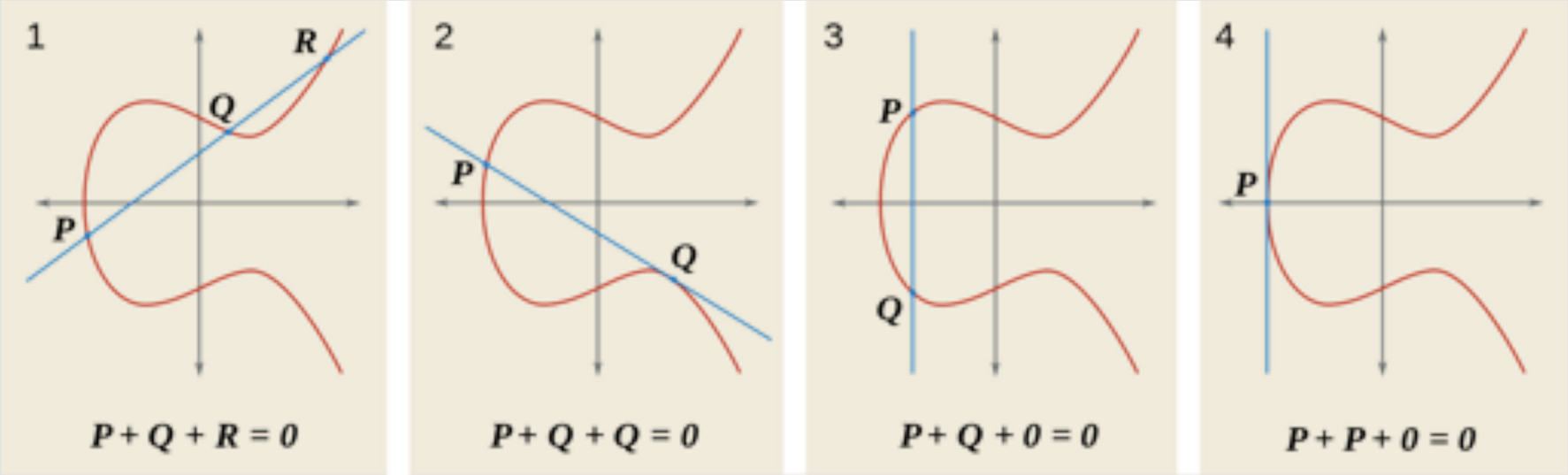
- The set of solutions (x,y) of an equation of the form $y^2=x^3+ax+b$ over a field with $4a^3 + 27b^2 \neq 0$. Also: $y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$.





$$\lambda = \frac{y_q - y_p}{x_q - x_p}$$
$$x_r = \lambda^2 - x_p - x_q$$
$$y_r = \lambda(x_p - x_r) - y_p$$

ALL CASES



- Every case has its own formula, depending on the field and the curve shape: GF(q) is different from GF(2^n) and GF(3^n).

NEUTRAL ELEMENT

- The neutral element is the point (∞,∞) at infinity.

- Theorem: Let **K** be a field and a,b in K. If $4a^3 + 27b^2 \neq 0$, the set of $(x, y) \in \mathbf{K}^2$ s.t. $y^2=x^3+ax+b$, together with (∞,∞) is an abelian group.

- To avoid ∞, one can use projective coordinates:
  - $(x, y, z) \mapsto (x/z, y/z)$
  - (0,1,0) is the neutral element
  - In projective coordinates, the group operation does not require inversions.
  - At the end of an exponentiation, one can switch to affine coordinates with one inversion (and two multiplications).

EFFICIENCY OF ELLIPTIC CURVE OPERATIONS

- Number of finite field operations

Curve shape, representation	DBL	ADD	mADD	mDBL	TPL	DBL+ADD
Short Weierstrass projective	11	14	11	8		
Short Weierstrass projective with a4=-1	11	14	11	8		
Short Weierstrass projective with a4=-3	10	14	11	8		
Short Weierstrass Relative Jacobian[1]	10	11	(7)	(7)		18
Tripling-oriented Doche–Icart–Kohel curve	9	17	11	6	12	
Hessian curve extended	9	12	11	9		
Hessian curve projective	8	12	10	6	14	
Jacobi quartic XYZ	8	13	11	5		
Jacobi quartic doubling-oriented XYZ	8	13	11	5		
Twisted Hessian curve projective	8	12	12	8	14	
Doubling-oriented Doche–Icart–Kohel curve	7	17	12	6		
Jacobi intersection projective	7	14	12	6	14	
Jacobi intersection extended	7	12	11	7	16	
Twisted Edwards projective	7	11	10	6		
Twisted Edwards Inverted	7	10	9	6		
Twisted Edwards Extended	8	9	8	7		
Edwards projective	7	11	9	6	13	
Jacobi quartic doubling-oriented XXYZZ	7	11	9	6	14	
Jacobi quartic XXYZZ	7	11	9	6	14	
Jacobi quartic XXYZZR	7	10	9	7	15	
Edwards curve inverted	7	10	9	6		
Montgomery curve	4			3		

THE ORDER OF AN ELLIPTIC CURVE

- Hasse's theorem (1936): $|\#E(\mathbb{F}_q) - (q+1)| \leq 2\sqrt{q}$

- Schoof's theorem (1985): $\#E(\mathbb{F}_q)$ can be computed in deterministic polynomial time.

➤ Point counting for random curves in practice:

  ➤ In large characteristic, the Schoof-Elkies-Atkin algorithm (SEA)

  ➤ In small characteristic, the algorithms of Satoh (2000) and Mestre (2001).

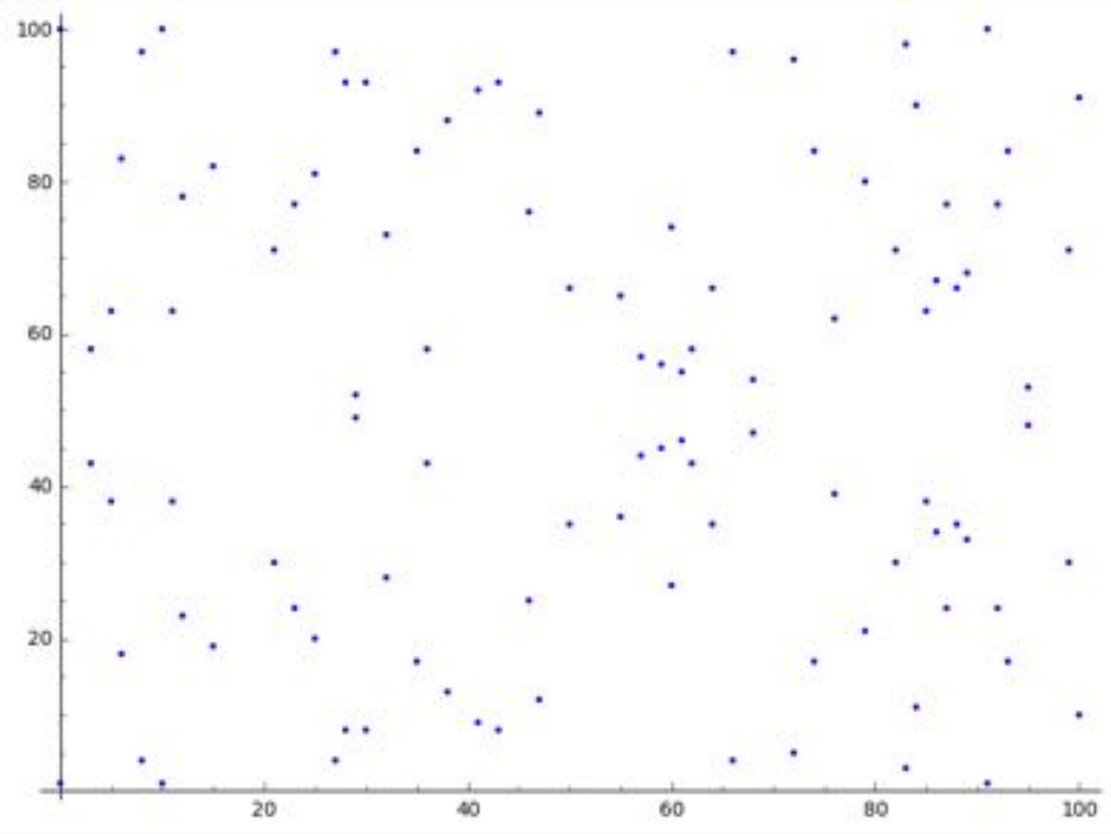
ELLIPTIC CURVES IN THE REAL WORLD

- 10% of Internet certificates use elliptic curves
  - 15 NSA curves standardized by NIST in 1999: P-384, P-512...
  - 2 independent standardized curves in 2019, used in Signal:
    - Curve25519 published in 2005 by Bernstein.
      $y^2 = x^3 + 486662x^2 + x \pmod{2^{255} - 19}$ where the starting point has x=9
    - Curve448 published in 2015 by Hamburg $y^2 + x^2 = 1 - 39081x^2y^2 \pmod{2^{448} - 2^{224} - 1}$
  - Bitcoin curve Secp256k1: $y^2 = x^3+7 \bmod 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$
  - France's curve: published in 2011, 77-digit prime number p, an integer b and two integers x and y s.t. $y^2=x^3-3x+b \pmod p$.

TOY EXAMPLE

- Ma courbe E est l'ensemble des triplets (x,y,z) mod 101 tels que

```
y^2 z = x^3 +  1 *x^2*z+ 1 *z^3 mod  101
Le nombre de points de la courbe vaut  105  =  3 * 5 * 7
```



ELLIPTIC CURVE CERTIFICATE

AffirmTrust Premium ECC
Autorité de certification racine
Expire le lundi 31 décembre 2040 15:20:24 heure normale d'Europe centrale
Ce certificat est valide

▶ **Se fier**
▼ **Détails**

Sujet	
Pays	US
Organisation	AffirmTrust
Nom	AffirmTrust Premium ECC
Nom de l'émetteur	
Pays	US
Organisation	AffirmTrust
Nom	AffirmTrust Premium ECC
Numéro de série	8401224907861490260
Version	3
Algorithme de signature	Signature ECDSA avec SHA-384 (1.2.840.10045.4.3.3)
Paramètres	aucun
Non valide avant	vendredi 29 janvier 2010 15:20:24 heure normale d'Europe centrale
Non valide après	lundi 31 décembre 2040 15:20:24 heure normale d'Europe centrale
Infos de clé publique	
Algorithme	Clé publique à courbe elliptique (1.2.840.10045.2.1)
Paramètres	Courbe elliptique secp384r1 (1.3.132.0.34)
Clé publique	97 octets : 04 0D 30 5E 1B 15 9D 03 …
Dimension de clé	384 bits
Utilisation de la clé	Vérifier
Signature	102 octets : 30 64 02 30 17 09 F3 87 …

STANDARD CURVES

- There are a few pre-computed curves with special points: the group is generated by the special point.
- secp384r1

```
p = 2^384-2^128-2^96+2^32-1  (384 bits)
a = - 3
b =
27580193559959705877849011840389048093056905856361568521428707301988689241309860865136260764883745107765439761230575

The point
G=(26247035095799689268623156744566981891852923491109213387815615900925518854738050089022388053975719786650872476732087,
83257109614890299855467512895201081792878530488613155947092059024805031988441922443864376039294733307808651162787l) has order
3940200619639447921227904010014361380507973927046544667946905279627659399113263569398956308152294913554433653942643  (384 bits)
```

SAGE

- p = 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffeffffffff0000000000000000ffffffff
- K = GF(p)
- a = K(0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffeffffffff0000000000000000fffffffc)
- b = K(0xb3312fa7e23ee7e4988e056be3f82d19181d9c6efe8141120314088f5013875ac656398d8a2ed19d2a85c8edd3ec2aef)
- E = EllipticCurve(K, (a, b))
- G = E(0xaa87ca22be8b05378eb1c71ef320ad746e1d3b628ba79b9859f741e082542a385502f25dbf55296c3a545e3872760ab7, 0x3617de4a96262c6f5d9e98bf9292dc29f8f41dbd289a147ce9da3113b5f0b8c00a60b1ce1d7e819d7a431d7c90ea0e5f)
- E.set_order(0xffffffffffffffffffffffffffffffffffffffffffffffffc7634d81f4372ddf581a0db248b0a77aecec196accc52973 * 0x1)
- print("p =",p)
- print("a = -",p-a)
- print("b =",b)
- print("G =",G)
- print("Ordre =",0xffffffffffffffffffffffffffffffffffffffffffffffffc7634d81f4372ddf581a0db248b0a77aecec196accc52973 * 0x1)

SECP384R1

```
p = 2^384-2^128-2^96+2^32-1  (384 bits)
a = - 3
b =
27580193559959705877849011840389048093056905856361568521428707301988689241309860865136260764883745107765439761230575

The point
G=(26247035095799689268623156744566981891852923491109213387815615900925518854738050089022388053975719786650872476732087,
83257109614890299855467512895201081792878530488613155947092059024805031998844192244386437603929473330780865116278 71) has order 39402006196394479212279040100143613805079739270465446667946905279627659\
3991132635693989563081522949135544336539426 43  (384 bits)

G+G =
(13621383085114665223611537069999249335994549661075979100866078813133013\
90679204654798639248640660900363360053616481 :
21933325650940841369538204578070064804451893403314136885642470114978241 1\
70633179043576249504748352841115137159204480 : 1)
(ordre-1)*G =
(26247035095799689268623156744566981891852923491109213387815615900925518\
85473805008902238805397571978665087247673208 7 :
31076295234905449226732288810623505625791886221604131073239087501765218 5\
71612451104608622327865990668783520461484448 : 1)
ordre*G =  (0 : 1 : 0)
```

EL GAMAL WITH SECP384R1

Alice : ma clef secrete est x =
6667422691722794500054185092005341297336732023910018100813774055955088348728751891730829256442807142126795642582866 et ma clef publique est y =
(1424584999262532297283253312461679953149670109970109820755114950660912966154277918542293151480494111196386876660460 :
1530369159792590036958732533169103934969683482093726689337186453676777564910516038695481727066974511407987636484208686 : 1)
Le messsage est le point m =
(3150806476909918594080647640240198520016739262958684673757565292569946273112190026358511629460831078868572132598668 7 :
3864668979104055792349379534884157563265535952365739880734783823366124516447151240930187324984346159951293860613026969 : 1)
Le chiffrement de m est : a =
(1282617893324431692981438094115606750854060477623108466658399777959737297129631074378248222506632984865304465033400 :
3817401960410187661784357066050988079070050356767760851199846958288734327494709886590577343527694659312994311279552222 : 1) et b =
(3768010278595860723592212493909415574659435676889631829335902897220153501626229597740724591501927413620894310366113 :
1324494962068054910123920460127539157310823965195600550849747970013106273350089156731103607481100811065193962689780808 : 1)
On dechiffre en
(3150806476909918594080647640240198520016739262958684673757565292569946273112190026358511629460831078868572132598668 7 :
3864668979104055792349379534884157563265535952365739880734783823366124516447151240930187324984346159951293860613026969 : 1)

BACKDOORED PRNG

- Dual_EC_DRBG is a PRNG released in 2005, and standardized by NIST in 2007

- Cryptographers noticed in 2006-07 that this PRNG may have backdoors: it used special constants, and if these constants were generated maliciously, then the person who generated the constants can attack the PRNG and most applications that use it.

BACKDOORED PRNG

- According to Snowden documents, the NSA created Dual_EC_DRBG and lobbied to make it a standard.

- According to Reuters, the NSA secretly gave $10 million to the RSA company in 2004 to force Dual_EC_DRBG to be the default generator for their BSAFE library. If so, this would allow NSA to decrypt any BSAFE traffic.

- According to the NY Times, the NSA spends $250 million/year to insert backdoors in software/hardware/standards: this is the Bullrun program.

BACKDOORED PRNG

- Three years ago, it was announced that Jupiter Networks routers suffered a dramatic cryptographic failure due to variants of Dual_EC_DRBG.

➤It is unknown if the person who can attack is the NSA or somebody else.

➤Once a backdoor is put, it can be subverted by somebody else.

THE NIST CURVES

- In 1999, NIST published 15 curves generated by the NSA, like P-384, P-512:

➤Some over GF(p): $y^2 \equiv x^3 - 3x + b \pmod{p}$

➤Some over GF(2^n): $y^2 + xy \equiv x^3 + x^2 + b \pmod{p}$

➤where b=Hash(seed)

NEW NIST CURVES

- In 2013, concerns were raised over the NIST curves.
- Added two curves in 2019 not generated by the NSA:

➤128-bit security: Curve25519 created in 2005.
$y^2 = x^3 + 486662x^2 + x \pmod{2^{255} - 19}$ where the starting point has x=9.

➤224-bit security: Curve448 created in 2015.
$y^2 + x^2 = 1 - 39081x^2y^2 \pmod{2^{448} - 2^{224} - 1}$
Both are used by the Signal protocol.

ANSSI-APPROVED CURVES

R18

Paramètres de courbes elliptiques pour le DLOG

Les paramètres de courbes elliptiques P256r1, P384r1 et P512r1 de la famille Brainpool, les paramètres de courbes elliptiques P-256, P-384 et P-521 du NIST et les paramètres de courbes Curve25519 et Curve448 sont recommandés.

Familles de courbes	Courbes	R/O	Notes
Brainpool [RFC5639]	BrainpoolP256r1	R	5.3.a
	BrainpoolP384r1	R	
	BrainpoolP512r1	R	
NIST [FIPS186] (voir Annexe D.1.2)	NIST P-256	R	5.3.a
	NIST P-384	R	
	NIST P-521	R	
IETF [RFC7748]	Curve25519	R	5.3.b
	Curve448	R	

IMPLEMENTATION ISSUES

- ECC is harder to implement than RSA

➤DL signatures have more implementation issues than RSA signatures: any leakage on the one-time key k is deadly.

➤Constant-time implementations are more difficult.

- Both RSA and DL are very sensitive to implementation and randomness issues.
- ECC is faster and smaller than RSA, except for RSA verification.

PERFORMANCES

```
                  sign    verify    sign/s verify/s
rsa  512 bits 0.000065s 0.000005s  15466.0 200529.7
rsa 1024 bits 0.000118s 0.000010s   8461.9 100561.2
rsa 2048 bits 0.000781s 0.000030s   1279.8  33738.8
rsa 3072 bits 0.002821s 0.000052s    354.5  19232.6
rsa 4096 bits 0.005631s 0.000087s    177.6  11436.3
rsa 7680 bits 0.050055s 0.000314s     20.0   3185.7
rsa 15360 bits 0.250000s 0.001129s      4.0    885.8
                  sign    verify    sign/s verify/s
dsa  512 bits 0.000100s 0.000068s   9965.7  14807.9
dsa 1024 bits 0.000169s 0.000126s   5900.6   7964.9
dsa 2048 bits 0.000379s 0.000308s   2640.3   3246.4
```

```
160 bits ecdsa (secp160r1)   0.0003s   0.0003s   3330.5   3638.8
 192 bits ecdsa (nistp192)   0.0004s   0.0003s   2845.2   2997.6
 224 bits ecdsa (nistp224)   0.0005s   0.0005s   1946.0   2209.2
 256 bits ecdsa (nistp256)   0.0000s   0.0001s  28751.0  10980.2
 384 bits ecdsa (nistp384)   0.0014s   0.0011s    737.6    888.7
 521 bits ecdsa (nistp521)   0.0031s   0.0022s    322.3    445.7
 163 bits ecdsa (nistk163)   0.0003s   0.0007s   2941.8   1485.7
 233 bits ecdsa (nistk233)   0.0005s   0.0009s   2212.2   1079.2
 283 bits ecdsa (nistk283)   0.0008s   0.0015s   1322.9    679.2
 409 bits ecdsa (nistk409)   0.0012s   0.0024s    811.3    419.1
 571 bits ecdsa (nistk571)   0.0026s   0.0055s    378.4    182.7
 163 bits ecdsa (nistb163)   0.0004s   0.0008s   2567.6   1305.3
 233 bits ecdsa (nistb233)   0.0005s   0.0010s   1948.9   1050.6
 283 bits ecdsa (nistb283)   0.0008s   0.0016s   1260.3    639.6
 409 bits ecdsa (nistb409)   0.0013s   0.0025s    783.7    396.4
 571 bits ecdsa (nistb571)   0.0027s   0.0055s    365.4    182.4
```

OpenSSL

NIST KEYSIZES

Table 2: Comparable strengths

Security Strength	Symmetric key algorithms	FFC (e.g., DSA, D-H)	IFC (e.g., RSA)	ECC (e.g., ECDSA)
≤ 80	2TDEA[21]	$L = 1024$ $N = 160$	$k = 1024$	$f = 160\text{-}223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$