

Elliptic Curve Crypto

in Practice [Bitcoin/Etc.]



Nicolas T. Courtois



- University College of London

Groups

Evariste Galois

Very famous French mathematician.

- At age of 14 started reading very serious books papers about algebra and mathematics.
- For reasons that are not fully explained failed all his exams to enter Ecole Polytechnique and most of his brilliant work was published and recognised only later.
- Did completely solve the problem of solvability of polynomial equations in one variable: “Galois Theory”.
- Was a political activist, against the king of France, frequently arrested and writing math papers while in prison.
- Died at the age of 20 after a fatal duel with an artillery officer, to some in the context of a broken love affair, to some stage-managed by the royalist fractions and the police.



He was the first to use the word **Group**.

Group - Definition

0. closure

A set M with an operation

$\bullet: M \times M \rightarrow M$ such that:

- 1) Operation \bullet is associative
 - 2) Has an identity element 1 .
 $1 \bullet a = a \bullet 1 = a$
 - 3) Each element a has an inverse called a^{-1} .
 $a^{-1} \bullet a = a \bullet a^{-1} = 1$
- semi-group

monoid

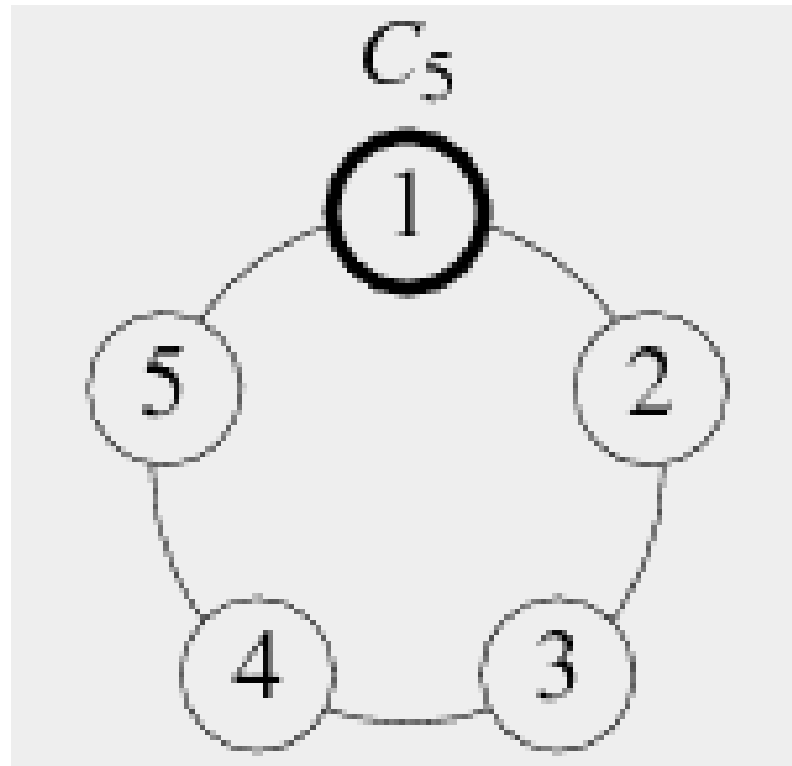
group

Abelian == Commutative Groups

$$a+b = b+a$$

Cyclic Groups

One element \Rightarrow generates the whole group.



Group – Example 1

Let $n \geq 2$.

$\{0, 1, 2, \dots, n-1\}, + \bmod n$ is a group.

Order of an Element

Order of an element g : $\text{ord}(g) =$

- the smallest integer such that $g^{\text{ord}(g)} = 1$.

Fermat's Little Theorem

Pierre de Fermat [1601-1655]:

French lawyer and government official,
one of the fathers of number theory
(also involved in breaking enemy ciphers and codes).



Theorem: Let p be a prime.

For any integer $a^p \equiv a \pmod{p}$.

Corollary: If $a \not\equiv 0 \pmod{p}$, then $a^{p-1} \equiv 1 \pmod{p}$.

Example 2.B.

$\{1,2,3\}$, $\ast \bmod 4$ is NOT a group.

Proof: 2 has no inverse.

Group ?

$\{1, 2, \dots, n-1\}$, $*$ mod n is a group IF AND ONLY IF n is a prime.

Rings

Rings

“When two operations work together nicely” like $+$ and $*$.

$(R, +, *, 0, 1)$ is a **Ring** if:

- $0 \neq 1$ (not serious, avoids one “trivial” ring $\{0\}, +, *$)
- $R, +$ is an Abelian group
- $R \setminus \{0\}, *$ is a monoid with identity element 1.
- $*$ distributes over $+$:
 $a(b+c) = ab+ac$
 $(b+c)a = ba+ca$

Fields

in modern usage always commutative

Fields [Abel, Galois]

2 added requirements:

- commutative
- each element $a \neq 0$ has an inverse

Corollary: When p is prime, \mathbb{Z}_p is a field.

Example 2.B.

$(\{0,1,2,3\}, + \bmod 4, * \bmod 4)$ is a ring.

It is NOT a field.

Why ?

Example 2.B.

$(\{0,1,2,3\}, + \bmod 4, * \bmod 4)$ is a ring.

It is NOT a field.

Why ?

Proof: **2** has no inverse.

Fields vs. Rings

- In a **field** we have all the 4 arithmetic operations $+, -, *, /$.
- In a **ring** we do not have $/$.

Finite Fields

Question:

$K = GF(p) = \mathbb{Z}_p$, p prime.

For certain polynomials, $\mathbb{Z}_p[X] / P(X)$ is a field.

Theorem: If and only if $P(X)$ is an irreducible polynomial.

Irreducible \Rightarrow has no proper divisor of lower degree.

Note: p is called the characteristic of this field.
 $x + x + \dots$ p times $= 0$.

Theorem:

ALL FINITE FIELDS are of the form $\mathbb{Z}_p[X] / P(X)$, with p prime.

Corollary: the number of elements of
a finite field is always $q=p^n$:

They are represented by all polynomials

$$a_0 + a_1 X^1 + \dots + a_{n-1} X^{n-1}.$$

corresponds to all possible n -tuples

$$(a_0, a_1, \dots, a_{n-1}).$$

Cycling

In \mathbb{Z}_p we had $a^p = a$ [Fermat's Little Thm.]

In any finite field F that has q elements
 $a^q = a$.

Theorem:

The multiplicative group of a finite field F is **cyclic**.

(in most cases false for \mathbb{Z}_n^* in general)

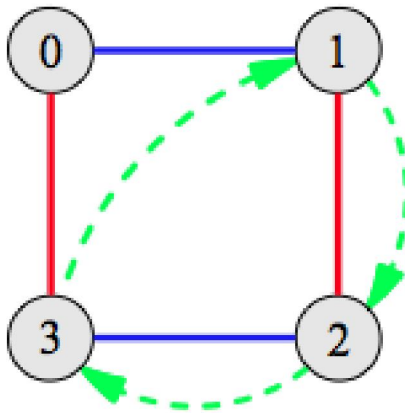
There is a generator element g , called **primitive element**, such that every element of the field $F \setminus \{0\}$ is a power of g .

(in fact there are MANY such elements).

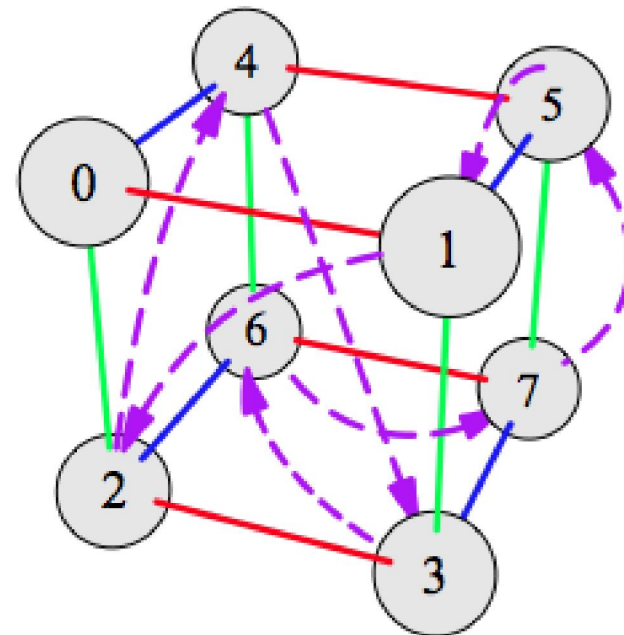
Theorem:

The multiplicative group of a finite field F is *cyclic*.

NOT OBVIOUS!



$GF(4)$



$GF(8)$

DL Problems

Various DL Problems

Discrete Logarithms

Given p , q , g and h find x such that

$$h = g^x \pmod{p}$$

where g is an element of order q in \mathbb{F}_p .

Elliptic Curve Discrete Logarithms

Given a curve E of order q over a field \mathbb{F}_p , and two points P and Q on the curve, find x such that

$$Q = [x]P.$$

*Slides from Jan 2013

Can take any (finite) group.

Bad Choices:

- ▶ Additive group \mathbb{Z} or \mathbb{F}_q .
- ▶ Multiplicative group of \mathbb{F}_q or \mathbb{C} .

Apparently Good Choices:

- ▶ Finite fields \mathbb{F}_q^* → is it OK?
- ▶ Elliptic curves over finite fields.
- ▶ Ideal class groups of number fields.
- ▶ Jacobian varieties of curves over finite fields.

Why NOT the cyclic group $GF(p^m)$?

find x such that

$$h = g^x \text{ in } GF(p^m)?$$

kind of broken...

DL in $\text{GF}(p^m)$ is Broken!

A quasi-polynomial algorithm for discrete
logarithm in finite fields of small characteristic

eprint/2013/400

Razvan Barbulescu¹, Pierrick Gaudry¹, Antoine Joux^{2,3}, and
Emmanuel Thomé¹

¹ Inria, CNRS, University of Lorraine, France

² Cryptology Chair, Foundation UPMC – LIP 6, CNRS UMR 7606, Paris, France

³ CryptoExperts, Paris, France

quasi-polynomial...

Abstract The difficulty of computing discrete logarithms in fields \mathbb{F}_{q^k} depends on the relative sizes of k and q . Until recently all the cases had a sub-exponential complexity of type $L(1/3)$, similar to the factorization problem. In 2013, Joux designed a new algorithm with a complexity of $L(1/4 + \epsilon)$ in small characteristic. In the same spirit, we propose in this article another heuristic algorithm that provides a quasi-polynomial complexity when q is of size at most comparable with k . By quasi-polynomial, we mean a runtime of $n^{O(\log n)}$ where n is the bit-size of the input. For larger values of q that stay below the limit $L_{q^k}(1/3)$, our algorithm loses its quasi-polynomial nature, but still surpasses the Function Field Sieve.

Elliptic Curves

Curves?

Curves: can be defined by sets of points which satisfy a certain systems of equations for example

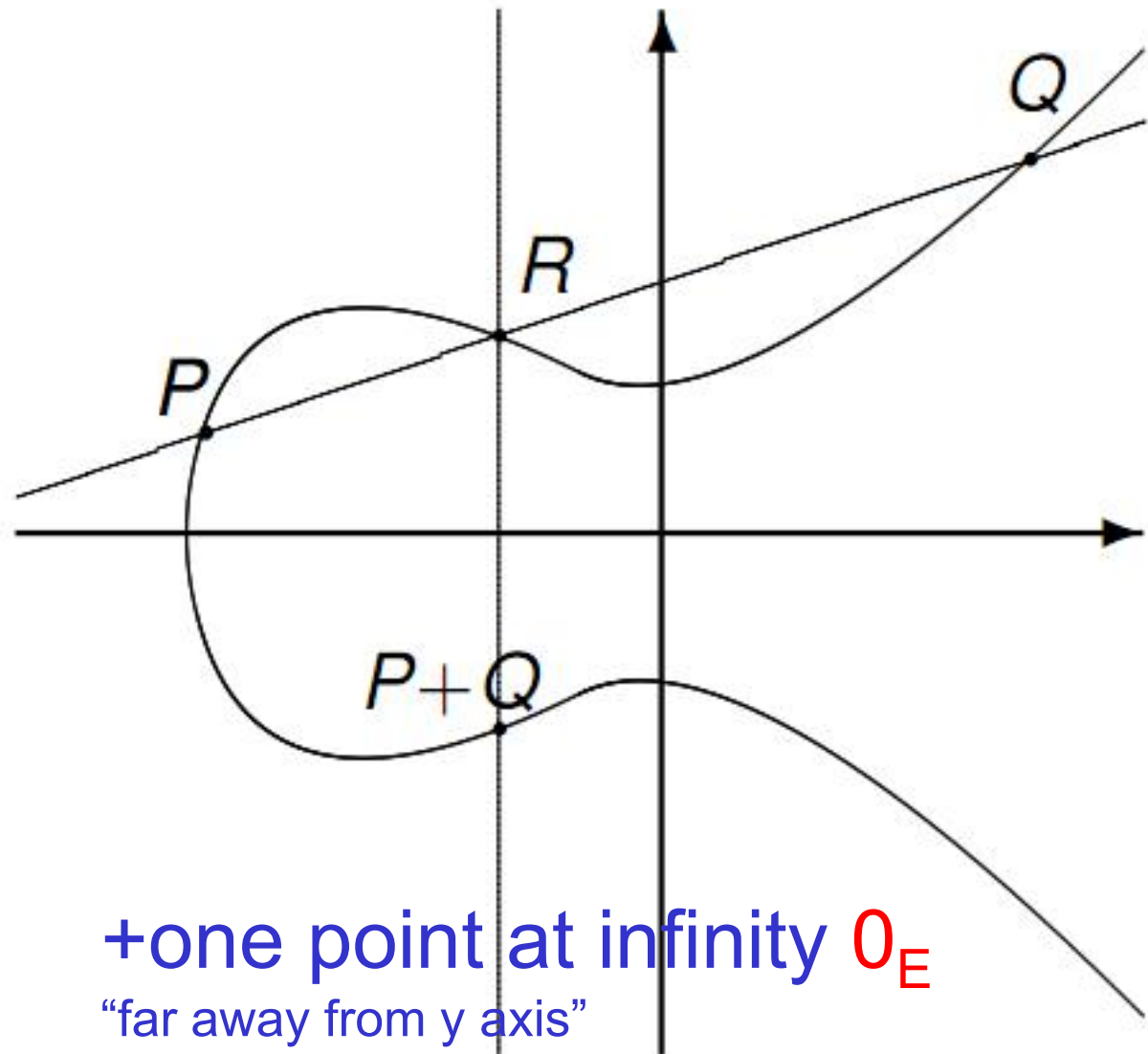
$$f(X,Y)=0 \dots$$

Curves Over Real Numbers

1 dim curve
in 2 dim space

some pairs (x,y)
belong
to the curve

In practice the do
NOT LOOK
like this!

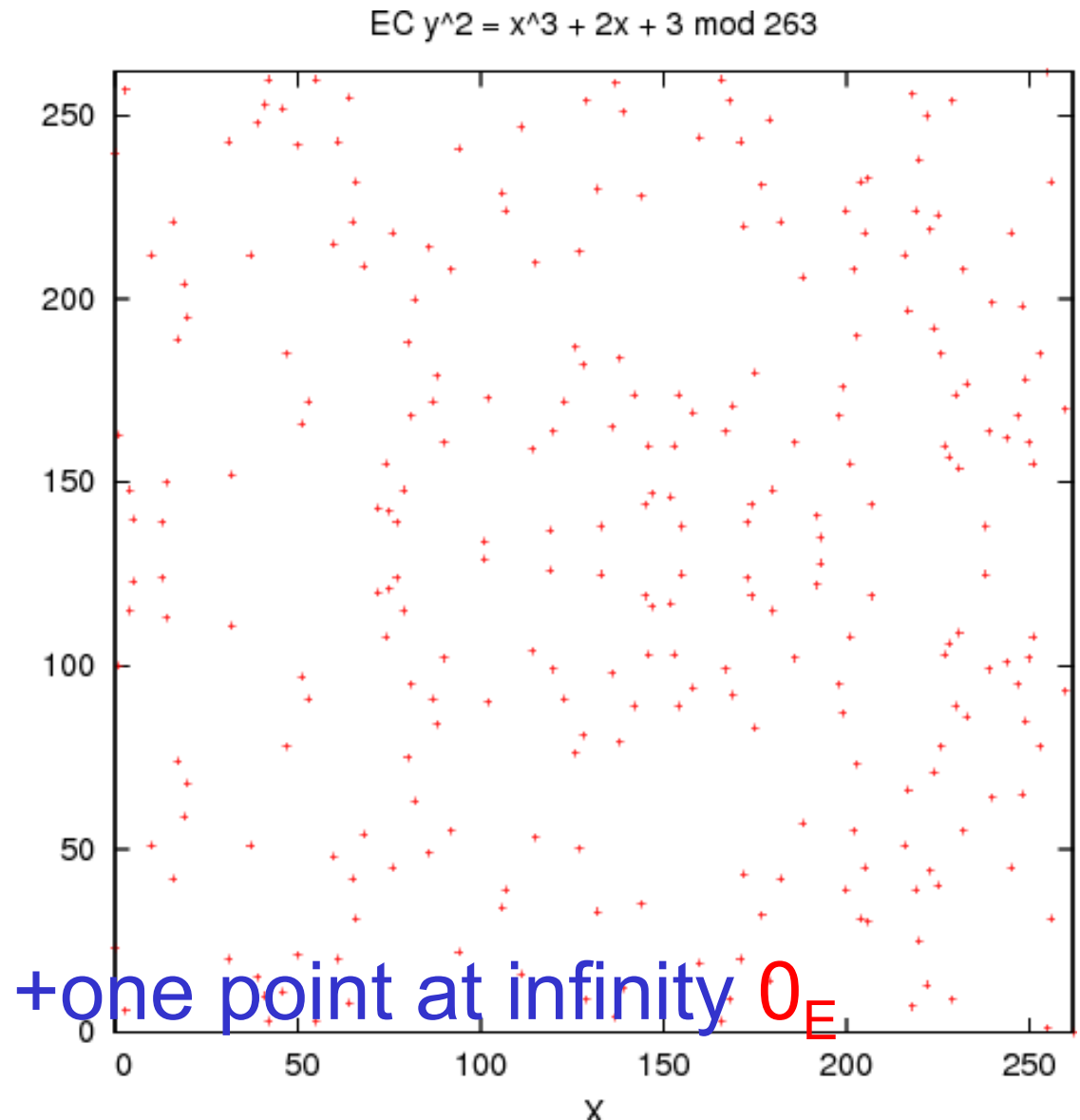


Over Finite Fields – Say Prime Fields

1 dim curve
in 2 dim space

some pairs (x,y)
belong
to the curve

THEY RATHER
LOOK like this!

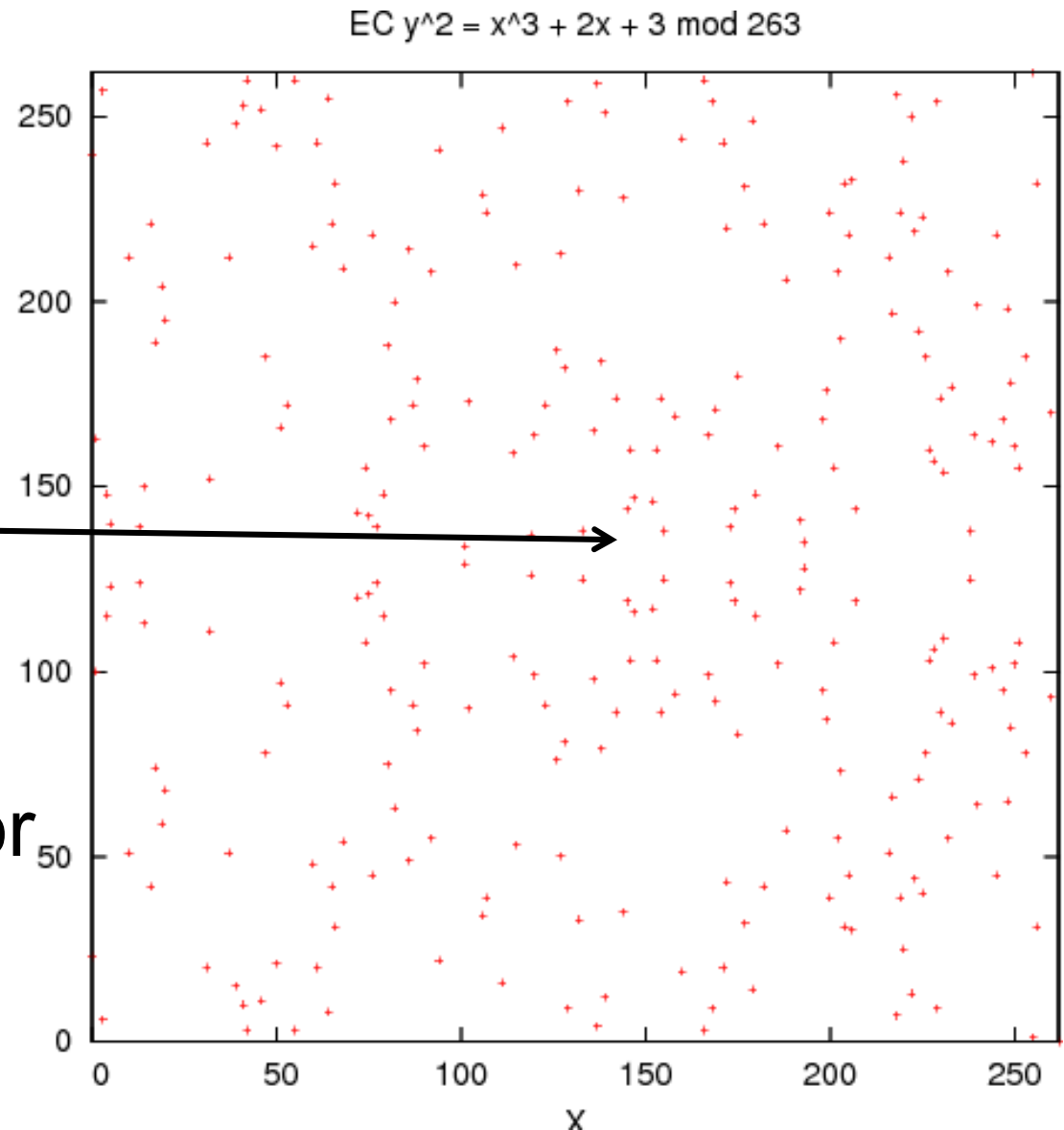


OWF Problem

We need a cyclic group
which is “complicated”
yet we can do
computations in it.

$x=0 \dots p-1$ $[x]G$

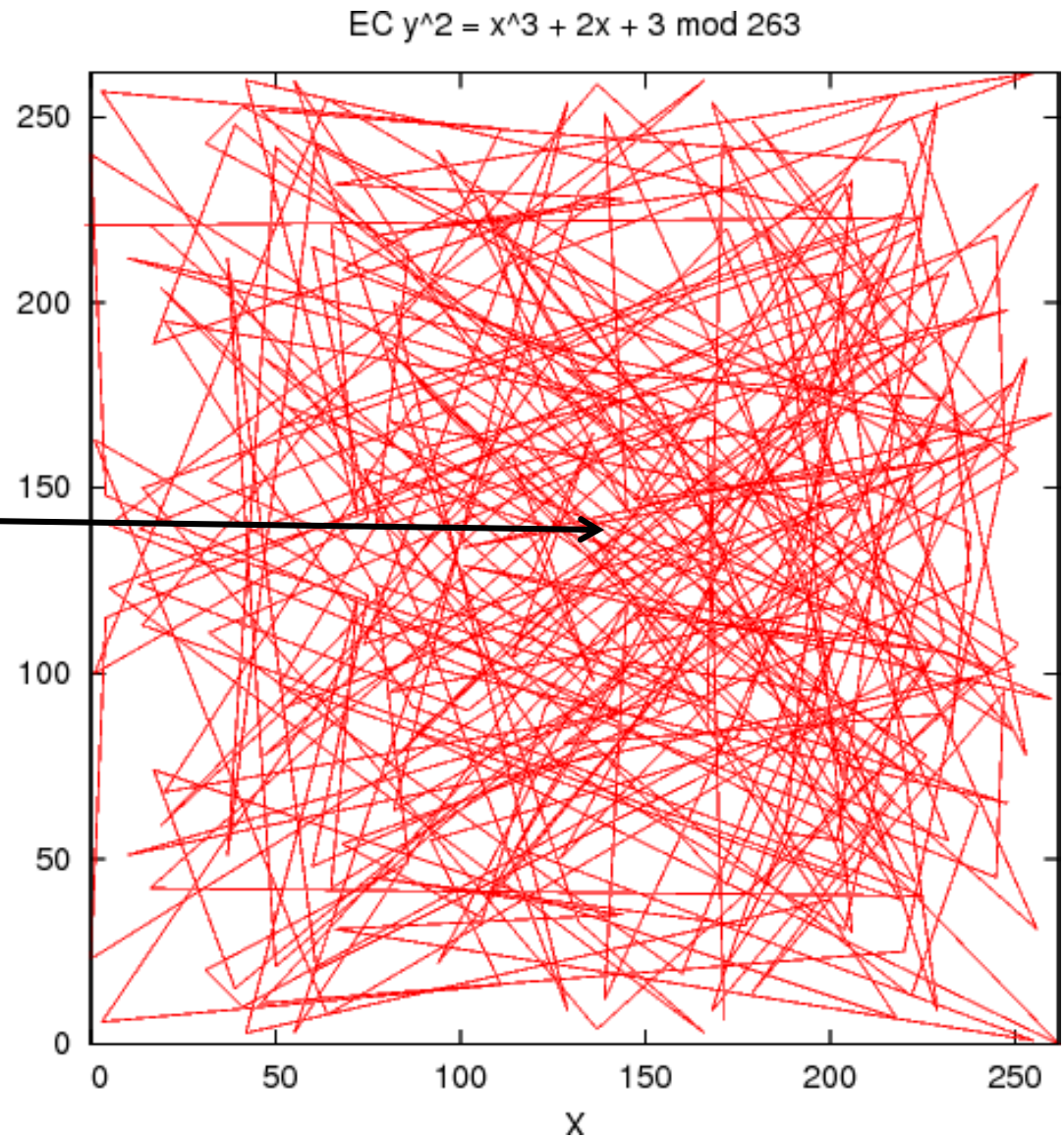
p = prime
 G = generator
point



OWF Problem

Straight Order (a cycle)
 \Rightarrow
“Complicated” Order

$x=0 \dots p-1$ $[x]G$



Elliptic Curves vs. RSA

Key Sizes

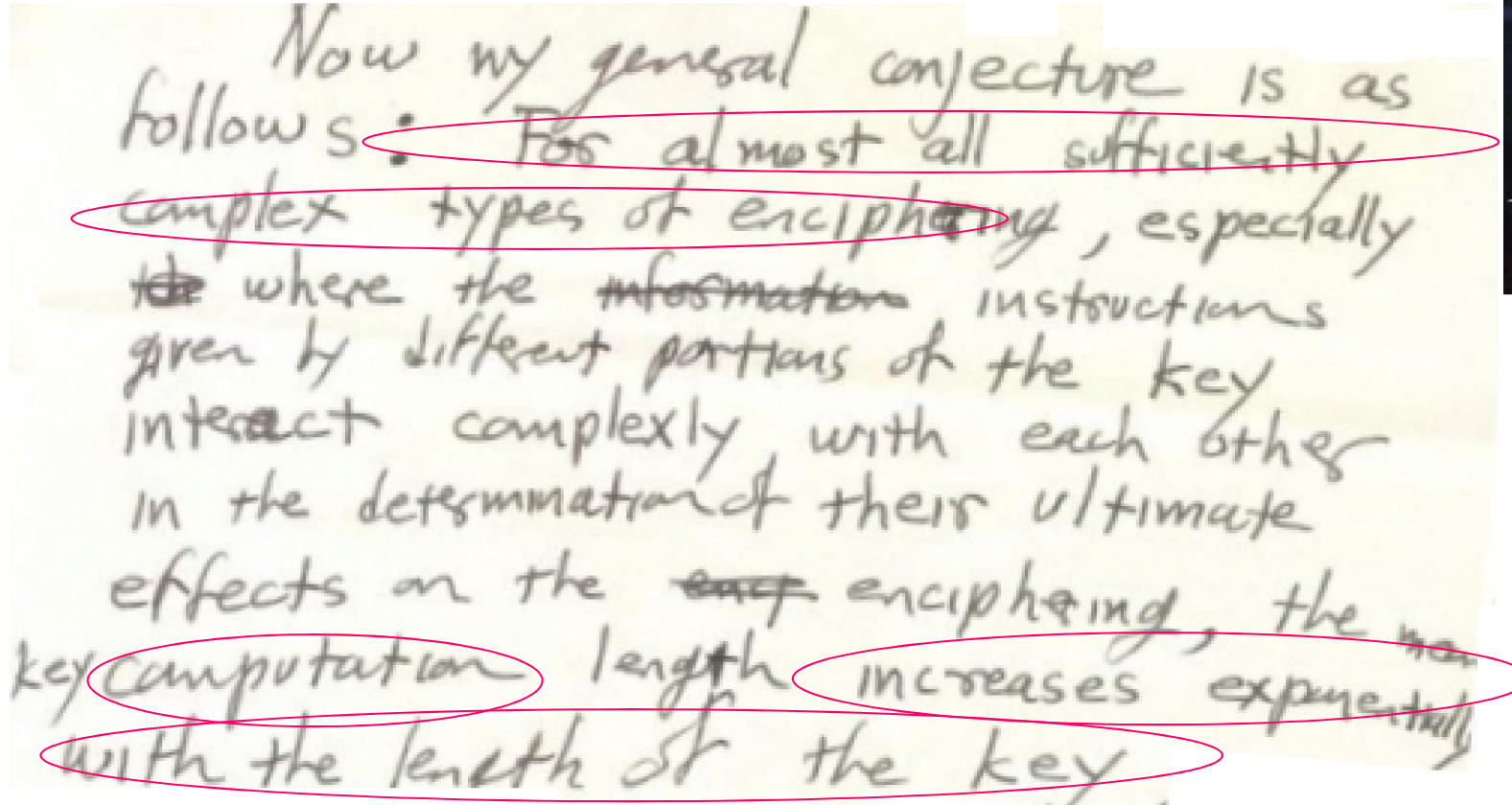
ECC Inventors

Neal Koblitz and Victor Miller,
independently, in 1985,

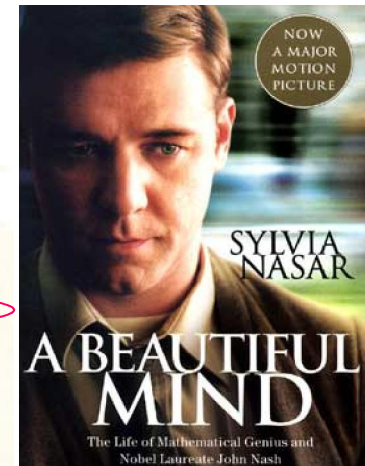
(proposed to do a DH in EC groups
=> whole of modern cryptography)

John Nash - 1955

In 2012 the NSA declassified his hand-written letter:



Now my general conjecture is as follows: For almost all sufficiently complex types of enciphering, especially ~~the~~ where the information instructions given by different portions of the key interact complexly with each other in the determination of their ultimate effects on the ~~enc~~ enciphering, the key computation length increases exponentially with the length of the key.

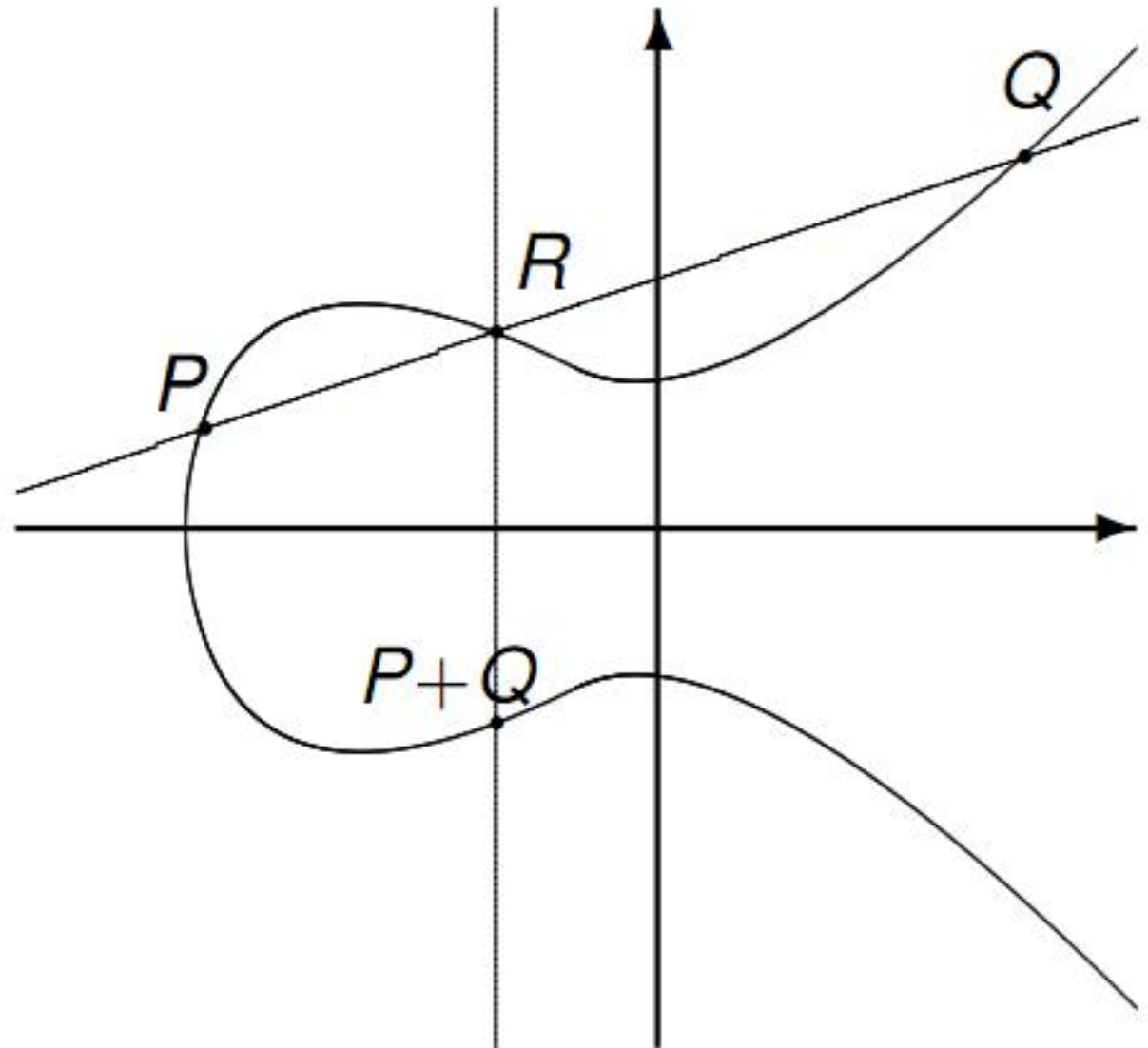
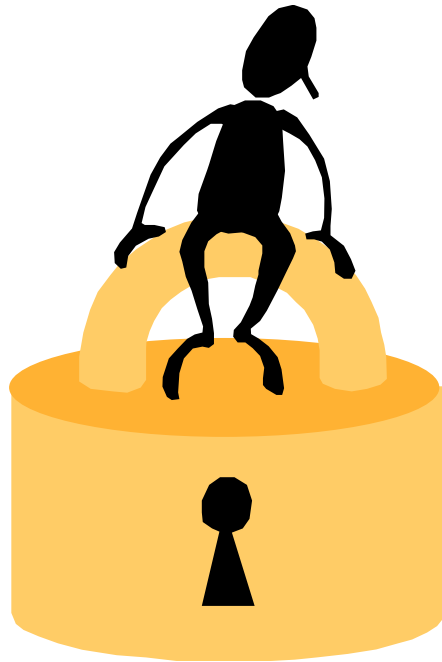


[...] “It means that it is quite **feasible to design ciphers that are effectively unbreakable**.

As ciphers become more sophisticated the game of cipher breaking by skilled teams, etc., should become a thing of the past.” [...] “The nature of this conjecture is such that I cannot prove it, even for a special type of ciphers. Nor do I expect it to be proven.”

Elliptic Curve Crypto

“exponential
security”



ECC! vs RSA

RSA: sub-exponential algos,
1024 bit keys provide only 80 bits of security with
NFS attack (a form of index calculus).

1986 Victor Miller claimed that:

“It is extremely unlikely that an ‘index calculus’ attack
on the elliptic curve method will ever be able to work.”

In V. Miller: “Use of elliptic curves in cryptography“, Crypto’85.

Best known attacks on general elliptic curves: $2^{n/2}$.

160 bit ECC key = 1024 bit RSA key? Not exactly but close:

256-bit ECC over prime field = 128-bit security

= secure for 50 years in absence of Quantum Computers

ECC/RSA Key Size Comparisons

(FIPS 186-2, Lenstra/Verheul, NESSIE)

Security level in bits	Block cipher	\mathbb{F}_p $ p $	\mathbb{F}_{2^m} m	RSA $ n $
80	SKIPJACK	192	163	1024
112	Triple-DES	224	233	2048
128	AES Small	256	283	3072
192	AES Medium	384	409	7680
256	AES Large	521	571	15360

ECC - Certicom Challenges [1997, revised 2009]

ECC2K-95	97	18322	\$ 5,000
ECC2-97	97	180448	\$ 5,000

ECCp-97	97	71982	\$ 5,000
---------	----	-------	----------

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECC2K-108	109	1.3×10^6	\$10,000
ECC2-109	109	2.1×10^7	\$10,000
ECC2K-130	131	2.7×10^9	\$20,000
ECC2-131	131	6.6×10^{10}	\$20,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECCp-109	109	9.0×10^6	\$10,000
ECCp-131	131	2.3×10^{10}	\$20,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECC2K-163	163	2.48×10^{15}	\$30,000
ECC2-163	163	2.48×10^{15}	\$30,000
ECC2-191	191	4.07×10^{19}	\$40,000
ECC2K-238	239	6.83×10^{26}	\$50,000
ECC2-238	239	6.83×10^{26}	\$50,000
ECC2K-358	359	7.88×10^{44}	\$100,000
ECC2-353	359	7.88×10^{44}	\$100,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECCp-163	163	2.3×10^{15}	\$30,000
ECCp-191	192	4.8×10^{19}	\$40,000
ECCp-239	239	1.4×10^{27}	\$50,000
ECCp-359	359	3.7×10^{45}	\$100,000

TOTAL = 725,000 USD

Bitcoin Elliptic Curve

secp256k1

Bitcoin EC

We define a finite field F_p with 256-bit prime $p =$

FFFFFFFF FFFFFFFFF FFFFFFFFF FFFFFFFFF FFFFFFFFF FFFFFFFFF FFFFFFFE FFFFC2F =
115792089237316195423570985008687907853269984665640564039457584007908834671663

Arithmetic modulo p is quite efficient. $p = 2^{256} - 2^{32} - 977$ ($977 = 2^9 + 2^8 + 2^7 + 2^6 + 2^4 + 1$).

The curve equation is $y^2 = x^3 + 7 \pmod{p}$.

(very special, not like “normal” elliptic curves, small integers only, one coeff=0)

Bitcoin EC

We define a finite field F_p with 256-bit prime $p =$
 115792089237316195423570985008687907853269984665640564039457584007908834671663

The curve equation is $y^2 = x^3 + 7 \pmod p$.

The base point G (generator) is: (could be any element)

in compressed form is:

$G = 02\ 79BE667E\ F9DCBBAC\ 55A06295\ CE870B07\ 029BFCDB\ 2DCE28D9\ 59F2815B\ 16F81798$

In uncompressed form it is: $G = 04$

$79BE667E\ F9DCBBAC\ 55A06295\ CE870B07\ 029BFCDB\ 2DCE28D9\ 59F2815B\ 16F81798$
 $483ADA77\ 26A3C465\ 5DA4FBFC\ 0E1108A8\ FD17B448\ A6855419\ 9C47D08F\ FB10D4B8$

or simply x, y are

$x = 55066263022277343669578718895168534326250603453777594175500187360389116729240$

$y = 32670510020758816978083085130507043184471273380659243275938904335757337482424$

Bitcoin EC

We define a finite field F_p with 256-bit prime $p=$
 115792089237316195423570985008687907853269984665640564039457584007908834671663

The curve equation is $y^2 = x^3 + 7 \pmod p$.

The base point G (generator) is:

$G = 02\ 79BE667E\ F9DCBBAC\ 55A06295\ CE870B07\ 029BFCDB\ 2DCE28D9\ 59F2815B\ 16F81798$

The order of G is $n=$

115792089237316195423570985008687907852837564279074904382605163141518161494337

another 256-bit prime such that $n \cdot G = 0$.

BTW. All the points on the curve lie in $\langle G \rangle$. $\#E(F_p) = n$

What's Wrong With Bitcoin Elliptic Curve?

ECC - Certicom Challenges [1997, revised 2009]

ECC2K-95	97	18322	\$ 5,000
ECC2-97	97	180448	\$ 5,000

ECCp-97	97	71982	\$ 5,000
---------	----	-------	----------

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECC2K-108	109	1.3×10^6	\$10,000
ECC2-109	109	2.1×10^7	\$10,000
ECC2K-130	131	2.7×10^9	\$20,000
ECC2-131	131	6.6×10^{10}	\$20,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECCp-109	109	9.0×10^6	\$10,000
ECCp-131	131	2.3×10^{10}	\$20,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECC2K-163	163	2.48×10^{15}	\$30,000
ECC2-163	163	2.48×10^{15}	\$30,000
ECC2-191	191	4.07×10^{19}	\$40,000
ECC2K-238	239	6.83×10^{26}	\$50,000
ECC2-238	239	6.83×10^{26}	\$50,000
ECC2K-358	359	7.88×10^{44}	\$100,000
ECC2-353	359	7.88×10^{44}	\$100,000

Challenge	Field size (in bits)	Estimated number of machine days	Prize (US\$)
ECCp-163	163	2.3×10^{15}	\$30,000
ECCp-191	192	4.8×10^{19}	\$40,000
ECCp-239	239	1.4×10^{27}	\$50,000
ECCp-359	359	3.7×10^{45}	\$100,000

secp256k1

NOT INCLUDED

no price if you
break it ☹

Koblitz citation:

"Once I heard a speaker from NSA complain about university researchers who are cavalier about proposing **untested cryptosystems**. He pointed out that in the real world if your cryptography fails, you lose a million dollars or your secret agent gets killed.

In academia, if you write about a cryptosystem and then a few months later find a way to break it, you've got two new papers to add to your résumé!"

Neal Koblitz,
Notices of the American Mathematical Society,
September 2007.

Official Bitcoin Wiki

https://en.bitcoin.it/wiki/Myths#Bitcoins_are_worthless_because_they_are_based_on_unproven_cryptography

“SHA256 and ECDSA which are used in Bitcoin are well-known industry standard algorithms. SHA256 is endorsed and used by the US Government and is standardized (FIPS180-3 Secure Hash Standard). If you believe that these algorithms are untrustworthy then you should not trust Bitcoin, credit card transactions or any type of electronic bank transfer.”

Bitcoin has a sound basis in well understood cryptography.

Well...actually it has major bug in it.

⇒ Major security scandal in the making?

⇒ Expect a lawsuit for

- failing to adopt the crypto/industry best practices
- and for supporting a dodgy cryptography standard
- and lack of careful/pro-active/ preventive security approach etc...
- Blame Satoshi ☺

k1 Promoters: Late Denial?

SECG = Standards for Efficient Cryptography group,
Industry consortium offspring of Canadian Certicom: the people
who proposed/promoted **secp256k1** in the first place.

Their document claims that both offer the same level of security=RSA-3072

=> not even strictly true in current literature, k1 has a slightly faster attack, cf. Cryptrec report by Smart and Galbraith

Timely denial:

Dan Brown, **the SECG chair** has written on 18 September 2013:

-“I did not know that BitCoin is using secp256k1.

I am **surprised** to see anybody use secp256k1 instead of secp256r1”,

<https://bitcointalk.org/index.php?topic=289795.80>

Comparison:

Used/recommended by:	secp256k1	secp256r1
Bitcoin, anonymous founder, no one to blame...	Y	
SEC Certicom Research	now surprised	Y
TLS, OpenSSL	Y, ever used???	Y 98.3% of EC
U.S. ANSI X9.63 for Financial Services	Y	Y
NSA suite B, NATO military crypto		Y
U.S. NIST		Y
IPSec		Y
OpenPGP		Y
Kerberos extension		Y
Microsoft implemented it in Vista and Longhorn		Y
EMV bank cards XDA [2013]		Y
German BSI federal gov. infosec agency, y=2015		Y
French national ANSSI agency beyond 2020		Y

TLS Adoption

7 % of TLS and 10% of SSH connections
use ECCs [December 2013].

98.3 % of these use [secp256r1](#).

(no data reported on [k1](#) counterpart, negligible or zero).

Source: Bos-Halderman-[Heninger](#)-Moore-Naehrig-
Wustrow, Paper= [eprint/2013/734](#), slides=ask me.

Remark: Many SSL libraries such GnuTLS.org support only a subset of what OpenSSL does,
and they don't support [k1](#).

Is k_1 Weak?

Weakness?

Bitcoin curve is characterized by the so called “small class number” which some researchers suspect to be less secure than general curves, see Sections 5.1 and 5.3 in

http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1029_report.pdf

All Koblitz curves have small class number.

In fact, the Koblitz curves in the SEC standard all have class number 1.

A STRONGER (more paranoid) requirement of having a large so called CM Field Discriminant D to be $|D| > 2^{100}$, cf. <http://safecurves.cr.yp.to/disc.html>

The bitcoin elliptic curve has the LOWEST $|D|$ of all known standardized elliptic curves, and therefore it is potentially the least secure.

Such curves allow “slight speedups” for discrete log attacks however “the literature does not indicate any mechanism that could allow further speedups”.



Wanna Bet?

Bitcoin Cryptography Broken in 2015

Category: [Bitcoin](#)

By  [NCourtois](#) ★★★★★

Description

The digital signature scheme of bitcoin with SHA256+secp256k1 ECDSA will be broken before 1 September 2015 by cryptography researchers. The attack should allow to forge digital signatures for at least a proportion of 1/1 million bitcoin users and steal money from them. It should be done faster than 2^{100} point additions total including the time to examine the data.

Decision Logic



bitcoin, cryptography, SHA256, ECDSA, ECDL, secp256k1

betmoose.com - Totally Anonymous Bets In BTC!

FEATURED

Bitcoin Cryptography Broken in 2015

Category: [Bitcoin](#) By [NCourtois](#) ★★★★★

Description

The digital signature scheme of bitcoin with SHA256+secp256k1 ECDSA will be broken before 1 September 2015 by cryptography researchers. The attack should allow to forge digital signatures for at least a proportion of 1/1 million bitcoin users and steal money from them. It should be done faster than 2^{100} point additions total including the time to examine the data.



SHA256, ECDSA, ECDL, secp256k1

YES

Volume:	₿ 0.140
# of Bets:	3

₿

PAYOUT	ROI
₿ 0.00	0%

* assumes current weight and volumes

Place Anonymously

NO

Volume:	₿ 0.189
# of Bets:	6

₿ 0.1

PAYOUT	ROI
₿ 0.14327	43.27%

* assumes current weight and volumes

Place Anonymously

Computations on Elliptic Curves

Follow The Picture

Formulas for
point
addition

char $p > 2$

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

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

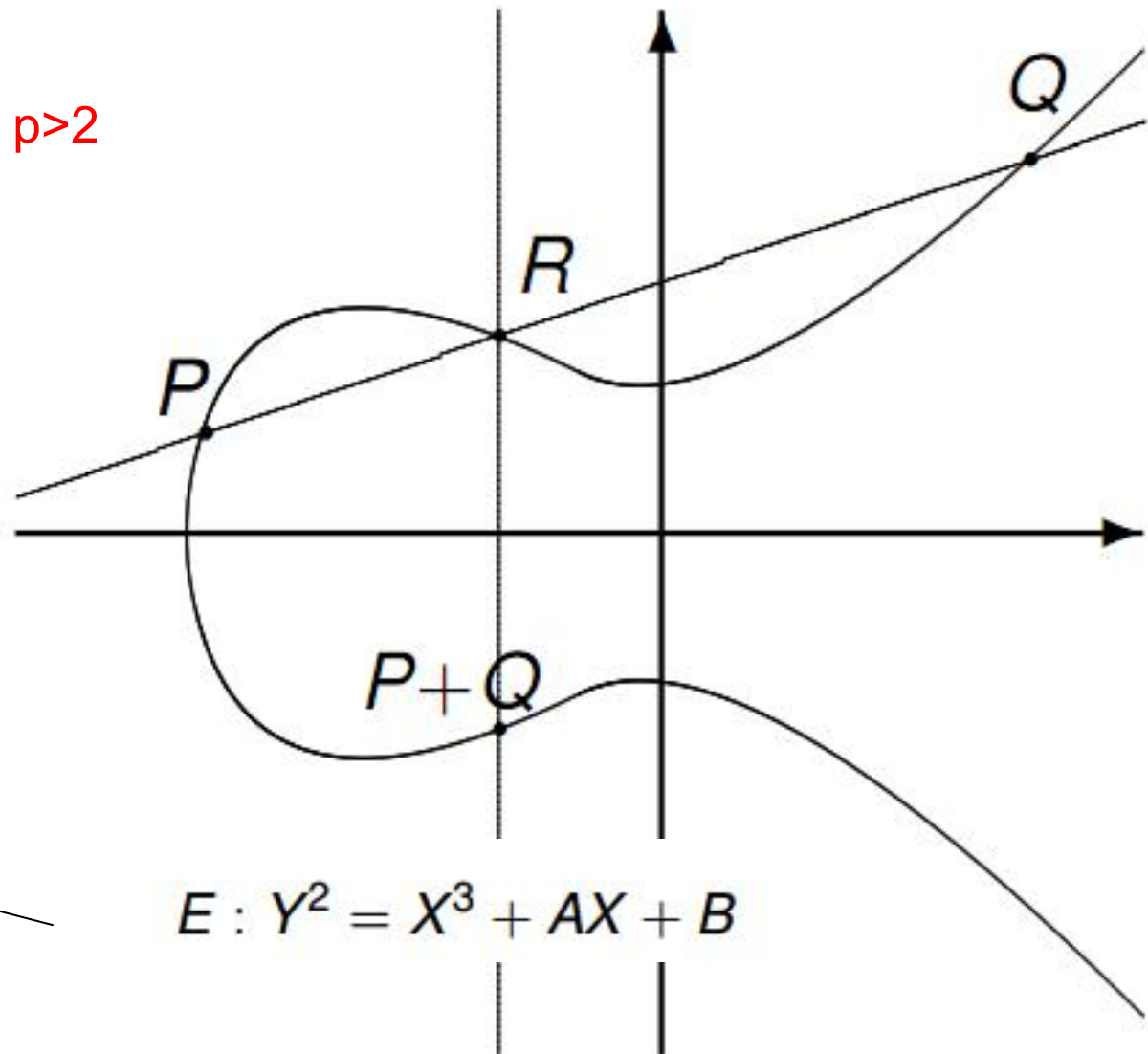
when $x_1 \neq x_2$ we set

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1},$$

~~when $x_1 = x_2$ and $y_1 \neq 0$ we set~~

~~see next slide~~

~~$$\lambda = \frac{3x_1^2 + A}{2y_1}$$~~



The Case of Doubling

Different
formula for
point
doubling

when $x_1 = x_2$ and $y_1 \neq 0$ we set

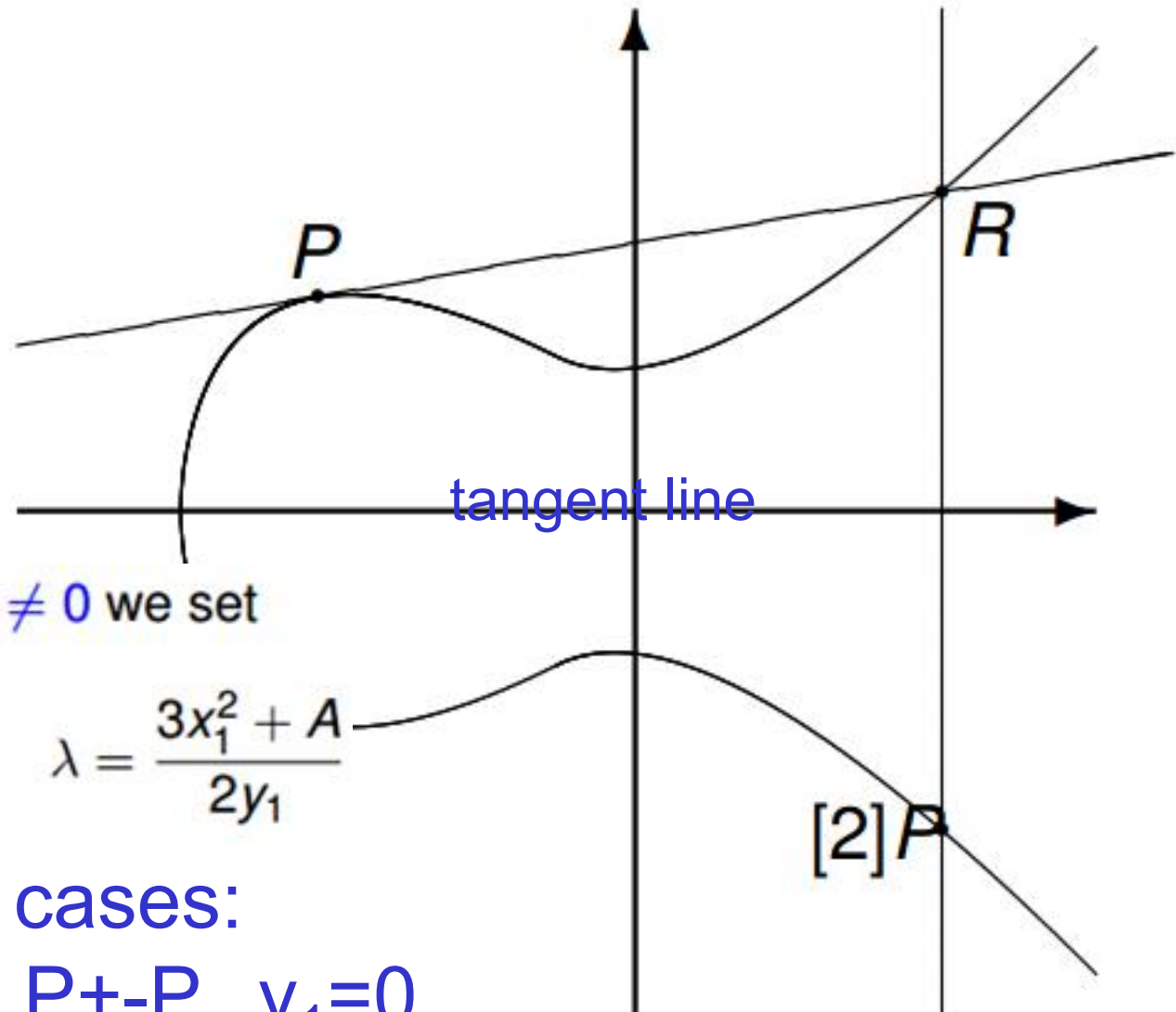
$$\lambda = \frac{3x_1^2 + A}{2y_1}$$

NOT a complete law!

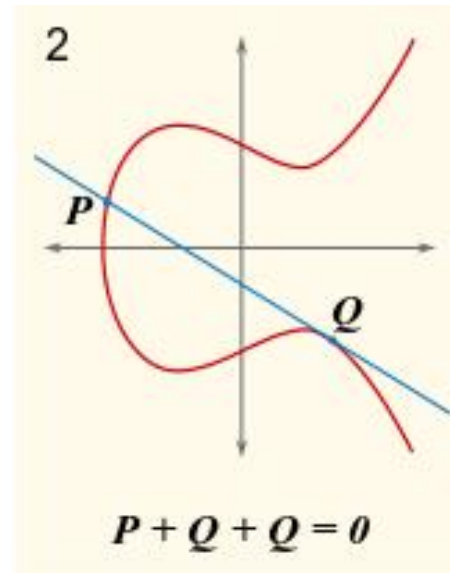
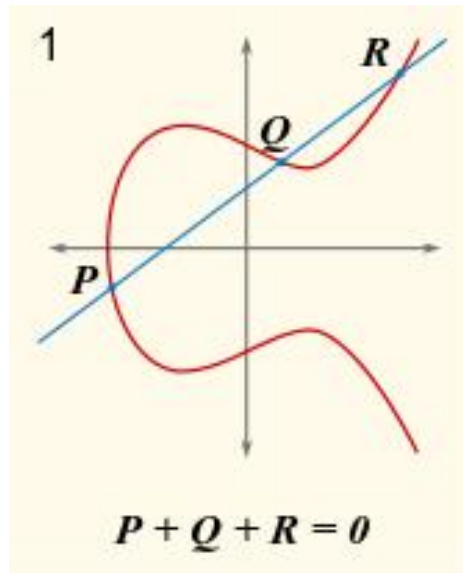
many special cases:

$0_E + 0_E, P + 0_E, P + -P, y_1 = 0$

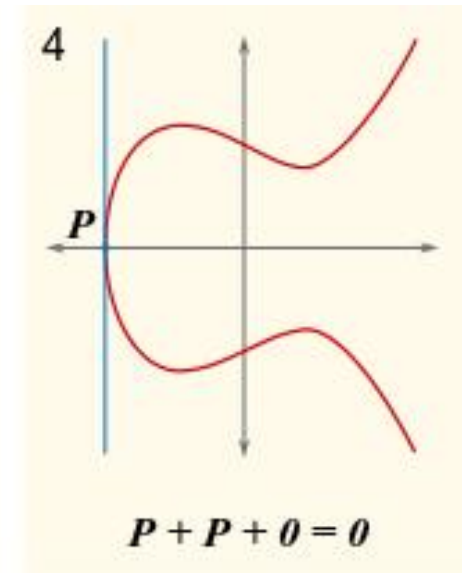
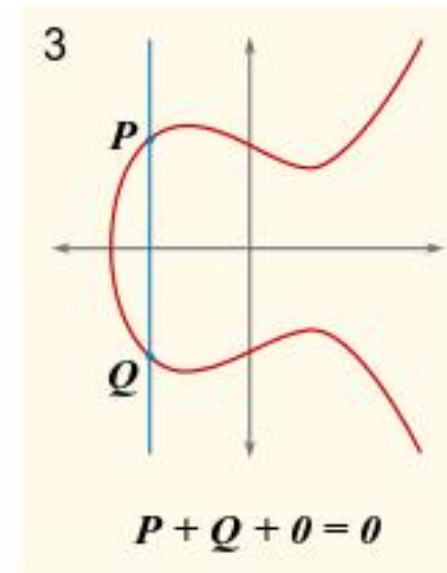
Nicolas T. Courtois, 2006



Special Cases => Not a “Complete Law”



tangent line



tangent line

RSA: exponentiation, modulo n

- e has $k = \log_2 n$ bits.

SQUARE AND MULTIPLY method:

Let $e = \sum_{i=1}^k 2^{i-1} e_i$ be the binary expansion of e .

- Compute $x, x^2, x^4, x^8, \dots, x^{2^{k-1}}$,
- Multiply all those for which $e_i = 1$.

Cost: about $k S + k/2 M$ on average.

ECC: exponentiation, 2 moduli p, n

- e has $k = \log_2 n$ bits.

DOUBLE AND ADD method:

Let $e = \sum_{i=1}^k 2^i e_i$ be the binary expansion of e .

- Compute $G, 2.G, 4.G, 8.G, \dots, (2^{k-1}).G$
- Add all those for which $e_i = 1$.

Cost: about $k \cdot \text{cost}(\text{double}) + k/2 \cdot \text{cost}(\text{add})$ on average.

=> Can further save on additions, with a bit more of memory
(store more multiples, use larger 'digits' than 1 bit)

Costs for ECC in “odd char” p

Point Addition:

- ▶ 6 Field Additions (Trivial)
- ▶ 3 General Field Multiplications
- ▶ 1 Field Inversion

F_p

$$3M+1I$$

$$1I=10-100M$$

Point Doubling:

- ▶ 5 Field Additions (Trivial)
- ▶ 2 Scalar/Field Multiplications (Trivial)
- ▶ 4 General Field Multiplications
- ▶ 1 Field Inversion

$$4M+1I$$

Projective Coordinates

$x, y \Rightarrow x, y, z$

avoids field inversions

Operation	Affine	Projective
Addition	$3M + 1I$	$16M$
Doubling	$4M + 1I$	$10M$

ECDSA

ECDSA

The Elliptic Curve Digital Signature Algorithm

- the elliptic curve analogue of DSA/DSS=outdated, 80-bit security.
- invented in 1992 by Scott Vanstone [died March 2014] in response to a NIST request for public comments on their first proposal for DSS

Adoption:

- accepted in 1998 in ISO 14888-3
- accepted in 1999 by ANSI X9.62 Financial Institutions standard.
- accepted in 2000 to become IEEE 1363-2000 standard
- accepted in 2000 as a NIST FIPS 186-2 standard.
- since 2002 also in ISO 15946-2
- since 2005 crucial part of NSA suite B (RSA was not included!)

ECDSA Signatures

Let d be a private key, integer mod n = ECC [sub-]group order.

- Pick a random non-zero integer $0 < a < n-1$.
- Compute $R = a \cdot P$, where P is the base point (generator).
- Let $r = (a \cdot P)_x$ be its x coordinate.
- Let $s = (H(m) + d \cdot r) / a \mod n$.

The signature of m is the pair (r, s) .

(512 bits in bitcoin)

***ECDSA Signature Generation

To sign a message m , A does the following:

1. Select a random integer k , $1 \leq k \leq n - 1$.
2. Compute $R = kP$ and $r = x(R) \bmod n$.
If $r = 0$ then go to step 1.
3. Compute $k^{-1} \bmod n$.
4. Compute $e = H(m)$, where H is a hash function.
5. Compute $s = k^{-1}(e + dr) \bmod n$. d=private key
If $s = 0$ then go to step 1.
6. A 's signature for the message m is (r, s) .

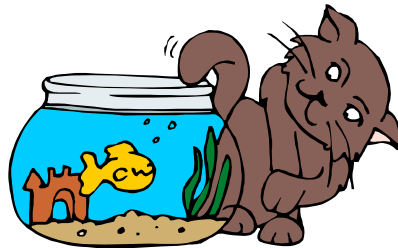
***ECDSA Signature Verif.

To verify A 's signature (r, s) on m , B should do the following:

1. Verify that r and s are integers in the interval $[1, n - 1]$.
2. Compute $e = H(m)$.
3. Compute $u_1 = es^{-1} \bmod n$ and $u_2 = rs^{-1} \bmod n$.
4. Compute $R = u_1P + u_2Q$ and $v = x(R) \bmod n$.
5. Accept the signature if and only if $v = r$.

$r, -s$ also valid

Dangers of ECDSA



Recall Sign. Generation

Let d be a private key, integer mod n = ECC [sub-]group order.

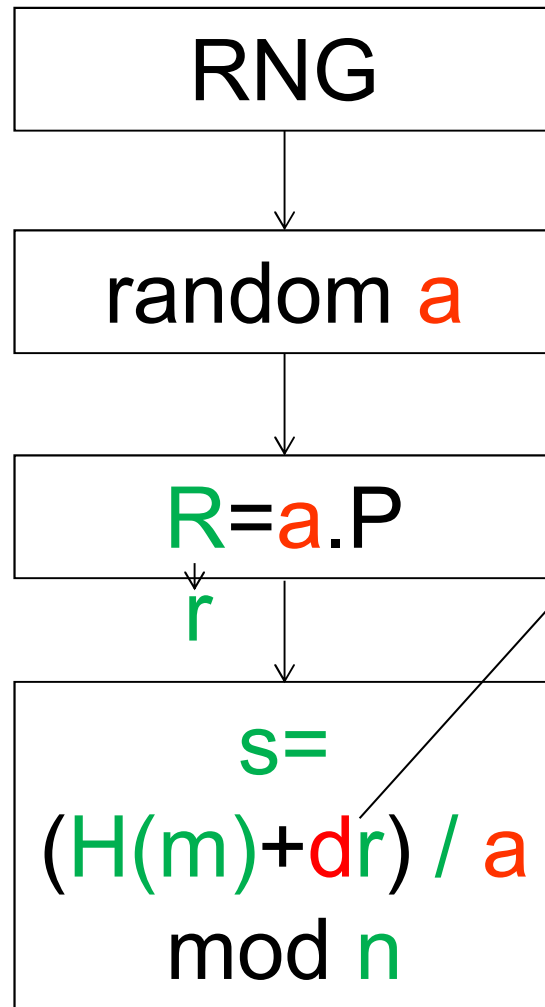
- Pick a random non-zero integer $0 < a < n-1$.
- Compute $R = a \cdot P$, where P is the base point (generator).
- Let $r = (a \cdot P)_x$ be its x coordinate.
- Let $s = (H(m) + d \cdot r) / a \mod n$.

The signature of m is the pair (r, s) .

(512 bits in bitcoin)

Attack Vectors (0)

random **a**: must be kept secret!



side channel
attacks/SPA/DPA
on the private key

d

(r, s)



Bad/Good RNG Attacks

1.

Bad RNG => the attacker CAN guess/brute force k . => recover private key

2.

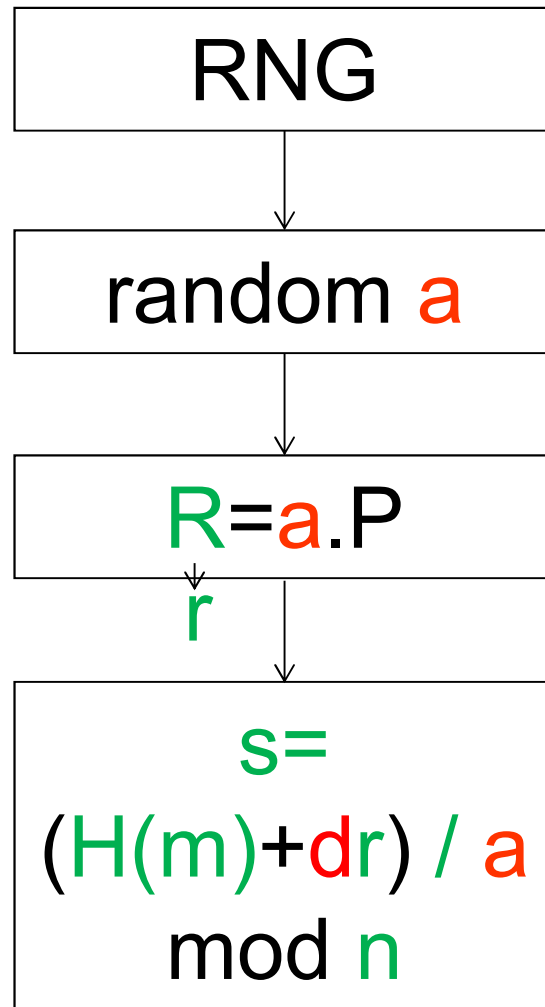
Bad RNG but attacker cannot guess it [e.g. obscurity]
=> there are still attacks (see next slides)!

3.

Good RNG but with side channels...

Key Problem

random **a**: should NEVER be revealed.



if **a** is revealed, the private key can be computed!

$$d = (sa - H(m)) / r \bmod n$$

(r,s)



Attack Vectors (1)

random **a**: must be kept secret!

RNG

random **a**

$R = \mathbf{a} \cdot P$

\mathbf{r}

$s =$
 $(H(m) + \mathbf{d} \mathbf{r}) / \mathbf{a}$
 $\text{mod } \mathbf{n}$

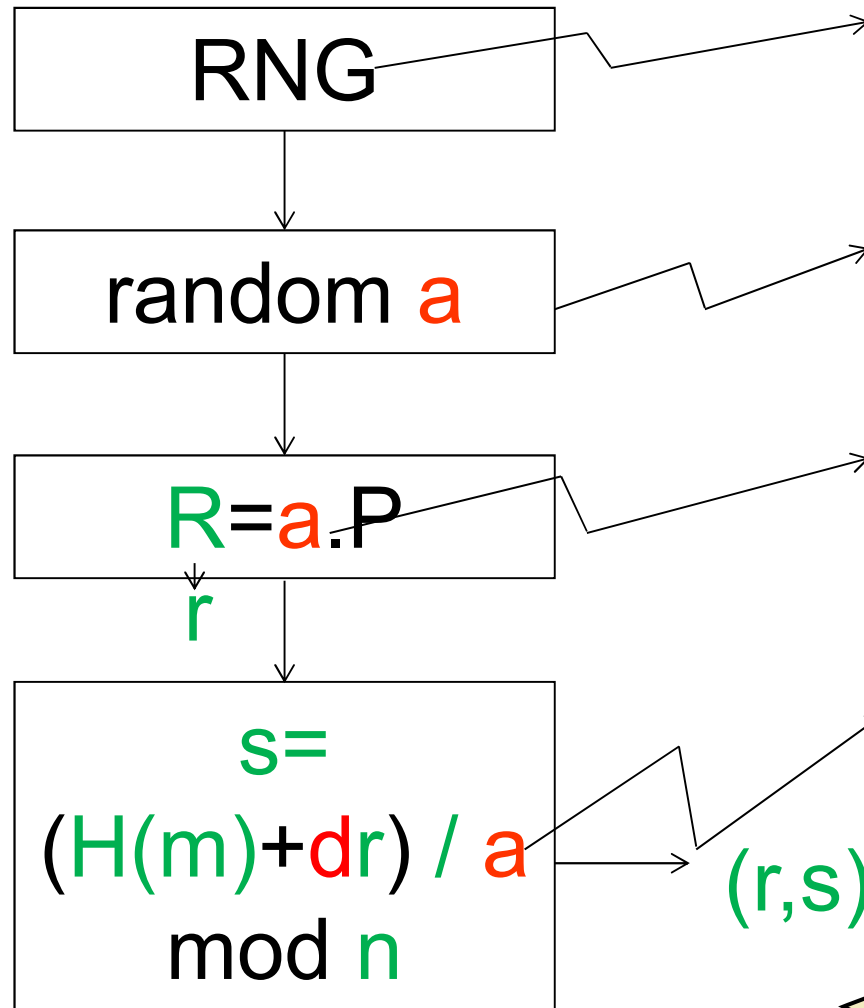
(\mathbf{r}, \mathbf{s})



used **/dev/random**
or **/dev/urandom**
or MsWin32
CryptGenRandom
but OS/CPU has
logged the seed!

Attack Vectors (2)

random **a**: must be kept secret!



side channel attack/SPA:
a USB drive powered by the PC has recorded the power consumption when generating/using **a**



Attack – 2 Users

random **a**: must be kept secret!

RNG

random **a**

$R = a \cdot P$

r

$s =$
 $(H(m) + dr) / a \pmod n$
 (r,s)

same **a** used twice =>

$$s_1 a = r d_1 + H(m_1) \pmod n$$

$$s_2 a = r d_2 + H(m_2) \pmod n$$

=>

$$r(s_2 d_1 - s_1 d_2) = s_2 H(m_1) - s_1 H(m_2)$$

has also happened
100s times in Bitcoin

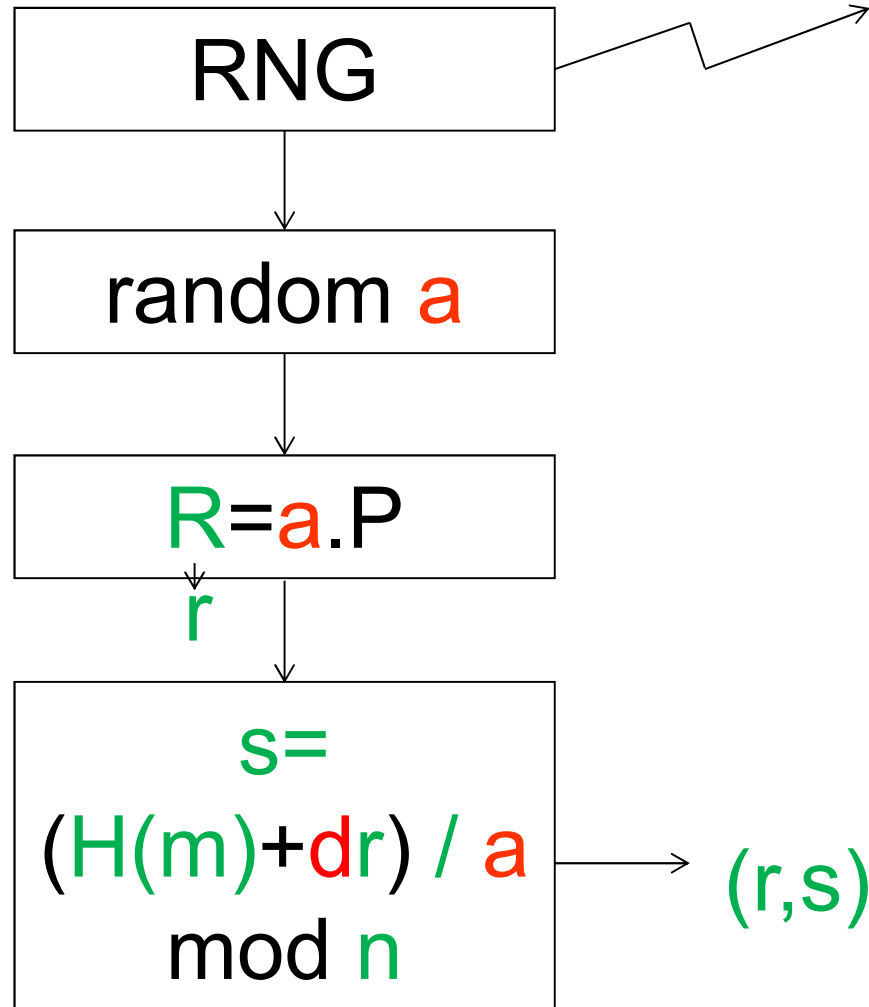
1 linear (affine) equation,
2 private keys “alone”

alone = no other secrets

each person can steal the
other person's bitcoins!

Attack Vectors (3)

random **a**: must be kept secret!



has also happened
100 times in Bitcoin

same **a** used twice by
the same user ($d_1 = d_2$).

In this case we have:

$$(s_1 a - H(m_1)) = r d = (s_2 a - H(m_2)) \bmod n$$

$$\Rightarrow a = (H(m_1) - H(m_2)) / (s_1 - s_2) \bmod n$$

AND now

$$d = (s a - H(m)) / r \bmod n$$

anybody can steal the bitcoins!

***Deterministic ECDSA!

Barwood-Wigley-Naccache-M'Raihi-Levy-dit-Vehel-Naccache-Pointcheval-Vaudenay-Katz-Wang etc...

Deterministic ECDSA

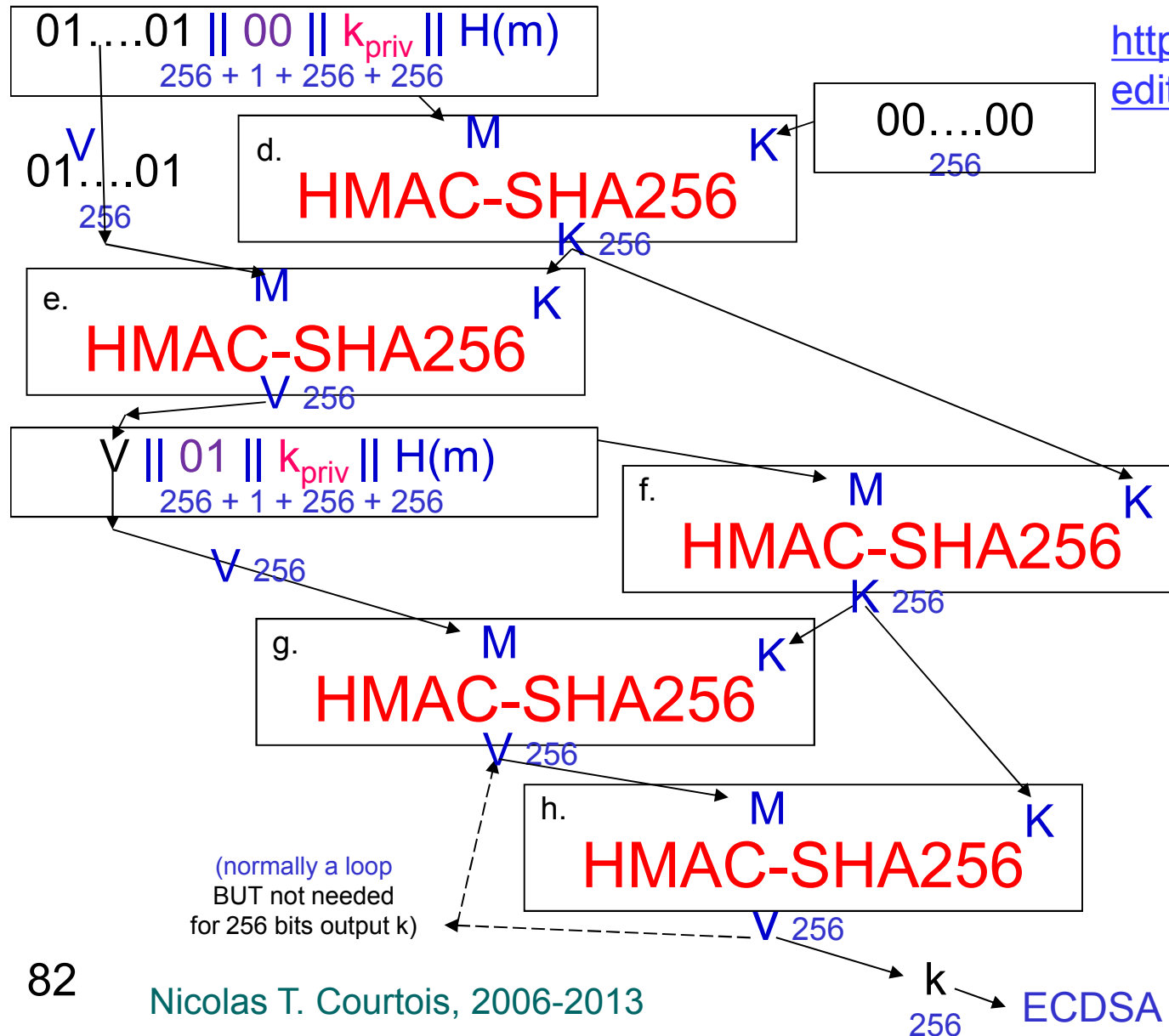
- Avoids attacks with bad RNG.
- Very strong protection against NSA backdoors such as hacking the RNG on the fly etc.
 - Deterministic => do it twice with different implementations, compare result.

Solution:

- RFC6979
- In pycoin library tx.py program:
 - uses a deterministic algorithm to create the ECSDA signatures, example to imitate.

RFC6979 [Pornin] = 5+ applications of HMAC

<http://www.rfc-editor.org/rfc/rfc6979.txt>



**HMAC-SHA256

Hashes twice with a key.

Definition (from [RFC 2104](#))

$$HMAC(K, m) = H((K \oplus opad) || H((K \oplus ipad) || m))$$

where

H is a cryptographic hash function,

K is a secret key padded to the right with the hash of the original key if it's longer than 160 bits,

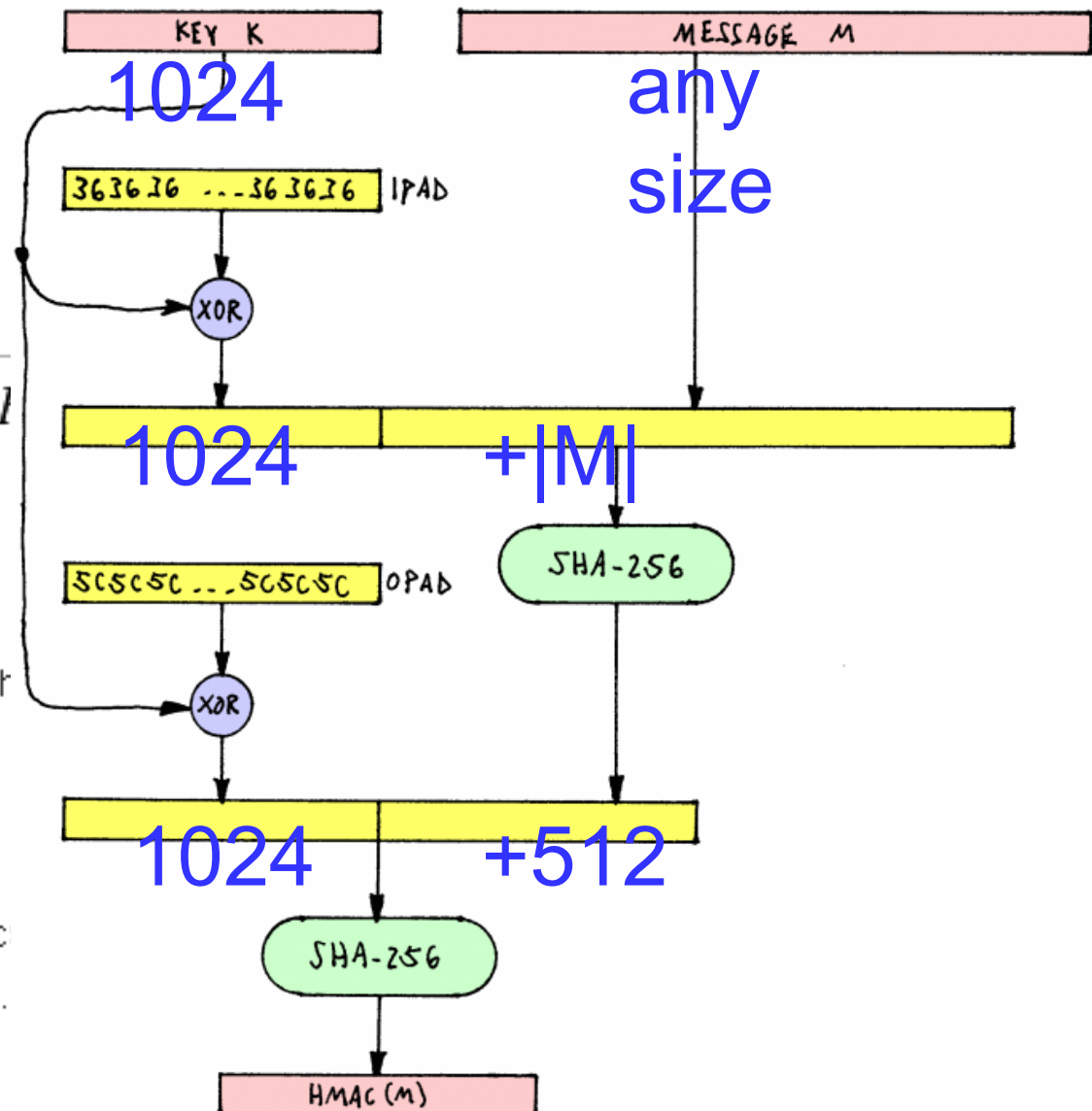
m is the message to be authenticated,

$||$ denotes concatenation,

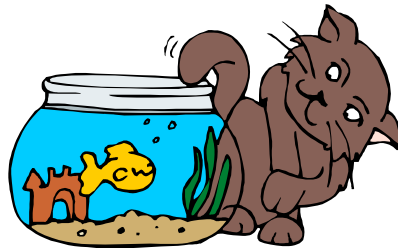
\oplus denotes exclusive or (XOR),

$opad$ is the outer padding (0x5c5c5c...5c

and $ipad$ is the inner padding (0x363636...



Bitcoin ECC Endomorphisms



Bitcoin EC

Base field = F_p with 256-bit prime $p = 2^{256} - 2^{32} - 977$

The curve equation is $y^2 = x^3 + 7 \pmod p$.

The base point G (generator) is:

$G = 02\ 79BE667E\ F9DCBBAC\ 55A06295\ CE870B07\ 029BFCDB\ 2DCE28D9\ 59F2815B\ 16F81798$

The order of G is $q =$

$115792089237316195423570985008687907852837564279074904382605163141518161494337$

another 256-bit prime such that $q \cdot G = 0$.

Special Multiples

Like “shortcuts in space”.

Fact: for the bitcoin elliptic curve
there exists MANY
special multiples (2 major ones in bitcoin)
such that:

$$\lambda * (x, y) = (\zeta * x, y)$$

3000 of μ s in general
100 μ s in bitcoin

0.2 μ s general curve
0.04 μ s bitcoin

Bad News:

There excessively few such multiples.

Remark 1. If λ works, λ^2 also works.

$$\lambda * (x, y) = (\zeta * x, y)$$

Bad news: We have

$$\lambda^6 = 1 \bmod q$$

There is exactly one non-trivial primitive root of degree 6, other are more or less the same like:

$$1, -1, \quad \lambda, -\lambda, \quad \lambda - 1 \equiv \lambda^2, -\lambda^2$$

Existing Solutions

There exists λ and ζ such that $(\forall (x,y) \in EC)$:

$$\lambda * (x, y) = (\zeta * x, y)$$

λ is a primitive 6-th root of unity mod q ,
 $\lambda^6 \equiv 1$ and $\lambda^3 \neq 1$ and $\lambda^2 \neq 1$

ζ is a primitive
6-th root of unity mod p

5363ad4cc05c30e0a5261c028812645a122e22ea2
0816678df02967c1b23bd73

7ae96a2b657c07106e64479eac3434e99cf
0497512f58995c1396c28719501ef

*Existing Solutions (twist 1)

There exists λ and ζ such that:

$$\lambda * (x, y) = (\zeta * x, -y)$$

λ is a primitive 6-th root of 1 mod q

ζ is a (no longer a primitive)
6-th root

*Existing Solutions (twist 2)

There exists λ' and ζ' such that:

$$\lambda' * (x, y) = (\zeta' * x, y)$$

λ' is a primitive 3rd root of 1

5363ad4cc05c30e0a5261c028812645a122
e22ea20816678df02967c1b23bd73-1

ζ' is a primitive 3rd root

7ae96a2b657c07106e64479eac3434e99cf
0497512f58995c1396c28719501ef-1

Extremely Few Such Points:

This probably at <http://safecurves.cr.yp.to/disc.html> we read:

Such curves allow “slight speedups” for discrete log attacks however
“the literature does not indicate any mechanism that could allow further
speedups”.

So until now this problem was not considered as very serious.