Bitcoin Signature

or

ECDSA on secp256k1

or

ElGamal with SHA on EC

S. Radziszowski  spre@cs.rit.edu
Nov 7, 2017
[Nov 28, 2017]
ECDSA - secp256k1

EC - elliptic curve cubic
F - field, f(x,y) - polynomial
  f - cubic in x, quadratic in y
E - set of (x,y), so f(x,y) = 0

DSA - digital signature algorithm
  as in NIST-DSS FIPS 1994, EC added in 2000

sec Standards for Efficient Crypto
  Certicom 2005, 2010

p256 F = Zp for special 256-bit prime p, p = \(2^{256}\)

k Koblitz
  almost so, but OK

l index (there is no 2, 3, ... )
Threads of this talk

1. Signatures

2. EC

3. special Bitcoin curve

4. security, no time ...

many sources:
textbooks
wiki
bitcoin developer guide
those missed will be listed in the next version of slides
Cryptosystem 7.5: Elliptic Curve Digital Signature Algorithm

Let \( p \) be a prime or a power of two, and let \( E \) be an elliptic curve defined over \( \mathbb{F}_p \). Let \( A \) be a point on \( E \) having prime order \( q \), such that the Discrete Logarithm problem in \( \langle A \rangle \) is infeasible. Let \( \mathcal{P} = \{0,1\}^*, \mathcal{A} = \mathbb{Z}_q^* \times \mathbb{Z}_q^* \), and define

\[ \mathcal{K} = \{(p,q,E,A,m,B) : B = mA\}, \]

where, \( 0 \leq m \leq q - 1 \). The values \( p, q, E, A \) and \( B \) are the public key, and \( m \) is the private key.

For \( K = (p,q,E,A,m,B) \), and for a (secret) random number \( k, 1 \leq k \leq q - 1 \), define

\[ \text{sig}_K(x, k) = (r, s), \]

where

\[ kA = (u, v) \]

\[ r = u \mod q, \quad \text{and} \]

\[ s = k^{-1}(\text{SHA}-1(x) + mr) \mod q. \]

(If either \( r = 0 \) or \( s = 0 \), a new random value of \( k \) should be chosen.)

For \( x \in \{0,1\}^* \) and \( r, s \in \mathbb{Z}_q^* \), verification is done by performing the following computations:

\[ w = s^{-1} \mod q \]

\[ i = w \text{SHA}-1(x) \mod q \]

\[ j = wr \mod q \]

\[ (u, v) = iA + jB \]

\[ \text{ver}_K(x, (r, s)) = \text{true} \iff u \mod q = r. \]
Basic Principle of Digital Signatures

Alice

$\rightarrow k_{pub}$

$x$

$s$

$(x, s)$

$k_{pub}$

ver

true / false

Bob

sig

$k_{pr}$
Definition 7.1: A signature scheme is a five-tuple \((\mathcal{P}, \mathcal{A}, \mathcal{K}, S, \mathcal{V})\), where the following conditions are satisfied:

1. \(\mathcal{P}\) is a finite set of possible messages
2. \(\mathcal{A}\) is a finite set of possible signatures
3. \(\mathcal{K}\), the keyspace, is a finite set of possible keys
4. For each \(K \in \mathcal{K}\), there is a signing algorithm \(\text{sig}_K \in \mathcal{S}\) and a corresponding verification algorithm \(\text{ver}_K \in \mathcal{V}\). Each \(\text{sig}_K : \mathcal{P} \to \mathcal{A}\) and \(\text{ver}_K : \mathcal{P} \times \mathcal{A} \to \{\text{true}, \text{false}\}\) are functions such that the following equation is satisfied for every message \(x \in \mathcal{P}\) and for every signature \(y \in \mathcal{A}\):

\[
\text{ver}(x, y) = \begin{cases} 
  \text{true} & \text{if } y = \text{sig}(x) \\
  \text{false} & \text{if } y \neq \text{sig}(x).
\end{cases}
\]

A pair \((x, y)\) with \(x \in \mathcal{P}\) and \(y \in \mathcal{A}\) is called a signed message.
Figure 9.15: Using a digital signature
Public-key System in Use
signature by hash and public-key encryption

**Signing**
- Data
  - Hash function
  - 101100110101
    - Hash
    - Encrypt hash using signer's private key
    - 111101101110
      - Signature
  - Certificate
  - Attach to data
- Digitally signed data

**Verification**
- Digitally signed data
  - 111101101110
    - Signature
    - Data
      - Hash function
      - Hash
        - 101100110101
          - ?
        - 101100110101
          - Hash
  - Decrypt using signer's public key

If the hashes are equal, the signature is valid.

[Wikipedia]
Cryptosystem 7.1: RSA Signature Scheme

Let \( n = pq \), where \( p \) and \( q \) are primes. Let \( \mathcal{P} = \mathcal{A} = \mathbb{Z}_n \), and define

\[
\mathcal{K} = \{ (n, p, q, a, b) : n = pq, p, q \text{ prime}, \ ab \equiv 1 \pmod{\phi(n)} \}.
\]

The values \( n \) and \( b \) are the public key, and the values \( p, q, a \) are the private key.

For \( K = (n, p, q, a, b) \), define

\[
\operatorname{sig}_K(x) = x^a \mod n
\]

and

\[
\operatorname{ver}_K(x, y) = \text{true} \iff x \equiv y^b \pmod{n}
\]

\((x, y \in \mathbb{Z}_n)\). 

slow

generating \( n \) is expensive
and cannot be shared by
different users
Cryptosystem 7.2: ElGamal Signature Scheme

Let $p$ be a prime such that the discrete log problem in $\mathbb{Z}_p$ is intractable, and let $\alpha \in \mathbb{Z}_p^*$ be a primitive element. Let $\mathcal{P} = \mathbb{Z}_p^*$, $\mathcal{A} = \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$, and define

\[ \mathcal{K} = \{(p, \alpha, \alpha, \beta) : \beta \equiv \alpha^a \pmod{p}\}. \]

The values $p$, $\alpha$ and $\beta$ are the public key, and $\alpha$ is the private key.

For $K = (p, \alpha, \alpha, \beta)$, and for a (secret) random number $k \in \mathbb{Z}_{p-1}^*$, define

\[ \text{sig}_K(x, k) = (\gamma, \delta), \]

where

\[ \gamma = \alpha^k \pmod{p} \]

and

\[ \delta = (x - \alpha \gamma)k^{-1} \pmod{(p - 1)}. \]

For $x, \gamma \in \mathbb{Z}_p^*$ and $\delta \in \mathbb{Z}_{p-1}$, define

\[ \text{ver}_K(x, (\gamma, \delta)) = \text{true} \iff \beta^\gamma \delta \equiv \alpha^x \pmod{p}. \]

\[ x = k\delta + \alpha \delta \]

\[ \alpha^x = \alpha^k \delta \]

1. can be forged for special $x$  
2. keep $k$ secret
Cryptosystem 7.3: Schnorr Signature Scheme

Let $p$ be a prime such that the discrete log problem in $\mathbb{Z}_p^*$ is intractable, and let $q$ be a prime that divides $p - 1$. Let $\alpha \in \mathbb{Z}_p^*$ be a $q$th root of 1 modulo $p$. Let $P = \{0, 1\}^*$, $A = \mathbb{Z}_q \times \mathbb{Z}_q$, and define

$$\mathcal{K} = \{(p, q, \alpha, a, \beta) : \beta \equiv \alpha^a \pmod{p}\},$$

where $0 \leq a \leq q - 1$. The values $p, q, \alpha$ and $\beta$ are the public key, and $a$ is the private key. Finally, let $h : \{0, 1\}^* \rightarrow \mathbb{Z}_q$ be a secure hash function.

For $K = (p, q, \alpha, a, \beta)$, and for a (secret) random number $k$, $1 \leq k \leq q - 1$, define

$$\text{sig}_K(x, k) = (\gamma, \delta),$$

where

$$\gamma = h(x \ || \ \alpha^k)$$

and

$$\delta = k + a\gamma \mod q.$$

For $x \in \{0, 1\}^*$ and $\gamma, \delta \in \mathbb{Z}_q$, verification is done by performing the following computations:

$$\text{ver}_K(x, (\gamma, \delta)) = \text{true} \Leftrightarrow h(x \ || \ \alpha^\delta \beta^{-\gamma}) = \gamma.$$
Cryptosystem 7.4: Digital Signature Algorithm

Let \( p \) be a \( L \)-bit prime such that the discrete log problem in \( \mathbb{Z}_p \) is intractable, where \( L \equiv 0 \pmod{64} \) and \( 512 \leq L \leq 1024 \), and let \( q \) be a 160-bit prime that divides \( p - 1 \). Let \( \alpha \in \mathbb{Z}_p^* \) be a \( q \)-th root of 1 modulo \( p \). Let \( \mathcal{P} = \{0, 1\}^* \), \( A = \mathbb{Z}_q^* \times \mathbb{Z}_q^* \), and define

\[
\mathcal{K} = \{(p, q, \alpha, \alpha, \beta) : \beta \equiv \alpha^a \pmod{p}\},
\]

where \( 0 \leq a \leq q - 1 \). The values \( p, q, \alpha \) and \( \beta \) are the public key, and \( a \) is the private key.

For \( K = (p, q, \alpha, a, \beta) \), and for a (secret) random number \( k, 1 \leq k \leq q - 1 \), define

\[
\text{sig}_K(x, k) = (\gamma, \delta),
\]

where

\[
\gamma = (\alpha^k \pmod{p}) \pmod{q} \quad \text{and} \quad \delta = (\text{SHA-1}(x) + a\gamma)k^{-1} \pmod{q}.
\]

(If \( \gamma = 0 \) or \( \delta = 0 \), a new random value of \( k \) should be chosen.)

For \( x \in \{0, 1\}^* \) and \( \gamma, \delta \in \mathbb{Z}_q^* \), verification is done by performing the following computations:

\[
e_1 = \text{SHA-1}(x) \delta^{-1} \pmod{q}
\]

\[
e_2 = \gamma \delta^{-1} \pmod{q}
\]

\[
\text{ver}_K(x, (\gamma, \delta)) = \text{true} \iff (\alpha^{e_1} \beta^{e_2} \pmod{p}) \pmod{q} = \gamma.
\]
Key generation

Key generation has two phases. The first phase is a choice of algorithm parameters which may be shared between different users of the system, while the second phase computes public and private keys for a single user.

Parameter generation

- Choose an approved cryptographic hash function \( H \). In the original DSS, \( H \) was always SHA-1, but the stronger SHA-2 hash functions are approved for use in the current DSS.\(^5\)\(^9\) The hash output may be truncated to the size of a key pair.

- Decide on a key length \( L \) and \( N \). This is the primary measure of the cryptographic strength of the key. The original DSS constrained \( L \) to be a multiple of 64 between 512 and 1,024 (inclusive). NIST 800-57 recommends lengths of 2,048 (or 3,072) for keys with security lifetimes extending beyond 2010 (or 2030), using correspondingly longer \( N \).\(^{10}\) FIPS 186-3 specifies \( L \) and \( N \) length pairs of (1,024, 160), (2,048, 224), (2,048, 256), and (3,072, 256).\(^{11}\) \( N \) must be less than or equal to the output length of the hash \( H \).

- Choose an \( N \)-bit prime \( q \).

- Choose an \( L \)-bit prime \( p \) such that \( p - 1 \) is a multiple of \( q \).

- Choose \( g \), a number whose multiplicative order modulo \( p \) is \( q \). This may be done by setting \( g = h^{(p - 1)/q} \mod p \) for some arbitrary \( h \) (\( 1 < h < p - 1 \)), and trying again with a different \( h \) if the result comes out as 1. Most choices of \( h \) will lead to a usable \( g \); commonly \( h = 2 \) is used.

The algorithm parameters \((p, q, g)\) may be shared between different users of the system.

Per-user keys

Given a set of parameters, the second phase computes private and public keys for a single user:

- Choose a secret key \( x \) by some random method, where \( 0 < x < q \).

- Calculate the public key \( y = g^x \mod p \).

There exist efficient algorithms for computing the modular exponentiations \( h^{(p - 1)/q} \mod p \) and \( g^x \mod p \), such as exponentiation by squaring.
Cryptosystem 7.5: *Elliptic Curve Digital Signature Algorithm*

Let \( p \) be a prime or a power of two, and let \( E \) be an elliptic curve defined over \( \mathbb{F}_p \). Let \( A \) be a point on \( E \) having prime order \( q \), such that the Discrete Logarithm problem in \( \langle A \rangle \) is infeasible. Let \( \mathcal{P} = \{0, 1\}^* \), \( \mathcal{A} = \mathbb{Z}_q^* \times \mathbb{Z}_q^* \), and define

\[
\mathcal{K} = \{ (p, q, E, A, m, B) : B = mA \},
\]

where \( 0 \leq m \leq q - 1 \). The values \( p, q, E, A \) and \( B \) are the public key, and \( m \) is the private key.

For \( K = (p, q, E, A, m, B) \), and for a (secret) random number \( k \), \( 1 \leq k \leq q - 1 \), define

\[
\text{sig}_K(x, k) = (r, s),
\]

where

\[
kA = (u, v)
\]

\[
r = u \mod q, \quad \text{and}
\]

\[
s = k^{-1} (\text{SHA}-1(x) + mr) \mod q.
\]

(If either \( r = 0 \) or \( s = 0 \), a new random value of \( k \) should be chosen.)

For \( x \in \{0, 1\}^* \) and \( r, s \in \mathbb{Z}_q^* \), verification is done by performing the following computations:

\[
w = s^{-1} \mod q
\]

\[
i = w \cdot \text{SHA}-1(x) \mod q
\]

\[
j = wr \mod q
\]

\[
(u, v) = iA + jB
\]

\[
\text{ver}_K(x, (r, s)) = \text{true} \iff u \mod q = r.
\]
The Generalized Discrete Logarithm Problem

- Given is a finite cyclic group $G$ with the group operation $\circ$ and cardinality $n$.

- We consider a primitive element $\alpha \in G$ and another element $\beta \in G$.

- The discrete logarithm problem is finding the integer $x$, where $1 \leq x \leq n$, such that:

\[
\beta = \underbrace{\alpha \circ \alpha \circ \cdots \circ \alpha}_{x \text{ times}} = \alpha^x
\]

or, in additive notation:

\[
x \int, \alpha, \beta \in G \\
\beta = \alpha + \alpha + \cdots + \alpha = x\alpha
\]

$x, \alpha \rightarrow x\alpha, \beta$ easy

$x, \alpha, \beta \rightarrow x$ infeasible to compute
The discrete logarithm problem in $\mathbb{Z}_p$

**Problem Instance**  \( I = (p, \alpha, \beta) \), where \( p \) is prime, \( \alpha \in \mathbb{Z}_p \) is a primitive element, and \( \beta \in \mathbb{Z}_p^* \).

**Objective**  Find the unique integer \( a \), \( 0 \leq a \leq p - 2 \), such that
\[
\alpha^a \equiv \beta \pmod{p}.
\]

We will denote this integer \( a \) by \( \log_\alpha \beta \).

---

**ECDL analog**

\[
I = (E, P, Q)
\]

\( E \) elliptic curve

\( P, Q \in E \), points

Find \( k \) such that \( Q = kP \)

\( k \) integer
Elliptic Curves over the Reals

Definition 6.3: Let $a, b \in \mathbb{R}$ be constants such that $4a^3 + 27b^2 \neq 0$. A non-singular elliptic curve is the set $E$ of solutions $(x, y) \in \mathbb{R} \times \mathbb{R}$ to the equation

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

(6.4)

together with a special point $\mathcal{O}$ called the point at infinity.
$y^2 = x^3 - x$
In cryptography, we are interested in elliptic curves module a prime $p$:

**Definition: Elliptic Curves over prime fields**

The elliptic curve over $\mathbb{Z}_p$, $p > 3$ is the set of all pairs $(x, y) \in \mathbb{Z}_p$ which fulfill

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

together with an imaginary point of infinity $\Theta$, where $a, b \in \mathbb{Z}_p$ and the condition

$$4a^3 + 27b^2 \neq 0 \mod p.$$ 

Note that $\mathbb{Z}_p = \{0, 1, \ldots, p - 1\}$ is a set of integers with modulo $p$ arithmetic.
\[4a^3 + 27b^2 = 0 \implies \text{singular EC}\]
\[\Downarrow \neq 0\]
6.4 has 3 different roots in \(\mathbb{C}\)

**Defining \(P + Q\)**

Suppose \(P, Q \in E\), where \(P = (x_1, y_1)\) and \(Q = (x_2, y_2)\). We consider three cases:

1. \(x_1 \neq x_2\)
2. \(x_1 = x_2\) and \(y_1 = -y_2\)
3. \(x_1 = x_2\) and \(y_1 = y_2\)

In case 1, we define \(L\) to be the line through \(P\) and \(Q\). \(L\) intersects \(E\) in the two points \(P\) and \(Q\), and it is easy to see that \(L\) will intersect \(E\) in one further point, which we call \(R'\). If we reflect \(R'\) in the \(x\)-axis, then we get a point which we name \(R\). We define \(P + Q = R\).

\(0 = \text{infinity}, \quad P + O = O + P = P\)
Point addition

trouble makers
(better not on EC over F)
Compuinations on Elliptic Curves (ctd.)

- Generating a group of points on elliptic curves based on point addition operation $P + Q = R$, i.e., $(x_P, y_P) + (x_Q, y_Q) = (x_R, y_R)$

- Geometric Interpretation of point addition operation
  - Draw straight line through $P$ and $Q$; if $P = Q$ use tangent line instead
  - Mirror third intersection point of drawn line with the elliptic curve along the x-axis

- Elliptic Curve Point Addition and Doubling Formulas

\[ x_3 = s^2 - x_1 - x_2 \mod p \quad \text{and} \quad y_3 = s(x_1 - x_3) - y_1 \mod p \]

where

\[ s = \begin{cases} 
  \frac{y_2 - y_1}{x_2 - x_1} \mod p & \text{if } P \neq Q \text{ (point addition)} \\
  \frac{3x_1^2 + a}{2y_1} \mod p & \text{if } P = Q \text{ (point doubling)}
\end{cases} \]
Computations on Elliptic Curves (ctd.)

The points on an elliptic curve and the point at infinity $\theta$ form cyclic subgroups

$2P = (5,1) + (5,1) = (6,3)$
$3P = 2P + P = (10,6)$
$4P = (3,1)$
$5P = (9,16)$
$6P = (16,13)$
$7P = (0,6)$
$8P = (13,7)$
$9P = (7,6)$
$10P = (7,11)$
$11P = (13,10)$
$12P = (0,11)$
$13P = (16,4)$
$14P = (9,1)$
$15P = (3,16)$
$16P = (10,11)$
$17P = (6,14)$
$18P = (5,16)$
$19P = \theta$

This elliptic curve has order $\#E = |E| = 19$ since it contains 19 points in its cyclic group.

$P = (5,1)$

Chapter 9 of Understanding Cryptography by Christof Paar and Jan Pelzl
Example 6.7  Let $E$ be the elliptic curve $y^2 = x^3 + x + 6$ over $\mathbb{Z}_{11}$.

\[11 = 3 \mod 4\]

\[\pm z^{(11+1)/4} \mod 11 = \pm z^3 \mod 11. = \sqrt{z} \mod 11\]

in action in secp256k1

<table>
<thead>
<tr>
<th>$z$</th>
<th>$x$</th>
<th>$x^3 + x + 6 \mod 11$</th>
<th>quadratic residue?</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>6</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>8</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>5</td>
<td>yes</td>
<td>4, 7</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>yes</td>
<td>5, 6</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>8</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>4</td>
<td>yes</td>
<td>2, 9</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>8</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>4</td>
<td>yes</td>
<td>2, 9</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>9</td>
<td>yes</td>
<td>3, 8</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>7</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>4</td>
<td>yes</td>
<td>2, 9</td>
</tr>
</tbody>
</table>

$\alpha = (2, 7) \quad 2\alpha = (5, 2) \quad 3\alpha = (8, 3)$

$4\alpha = (10, 2) \quad 5\alpha = (3, 6) \quad 6\alpha = (7, 9)$

$7\alpha = (7, 2) \quad 8\alpha = (3, 5) \quad 9\alpha = (10, 5)$

$10\alpha = (8, 8) \quad 11\alpha = (5, 9) \quad 12\alpha = (2, 4)$
PointCompress\((P) = (x, y \mod 2)\), where \(P = (x, y) \in E\).

PointCompress : \(E \backslash \{0\} \rightarrow \mathbb{Z}_p \times \mathbb{Z}_2\).

Algorithm 6.4: PointDecompress\((x, i)\)

\[
z \leftarrow x^3 + ax + b \mod p
\]

if \(z\) is a quadratic non-residue modulo \(p\)
then return ("failure")

\[
y \leftarrow \sqrt{z} \mod p
\]

if \(y \equiv i \pmod{2}\)
then return \((x, y)\)

else return \((x, p - y)\)


HP: US patent 6252960 B1 1998
expires in 2018

130+ crypto and EC patents:
NSA, Certicom, RSA Security, HP, Harris
Cryptosystem 7.5: Elliptic Curve Digital Signature Algorithm

Let $p$ be a prime or a power of two, and let $E$ be an elliptic curve defined over $\mathbb{F}_p$. Let $A$ be a point on $E$ having prime order $q$, such that the Discrete Logarithm problem in $(A)$ is infeasible. Let $\mathcal{P} = \{0, 1\}^*$, $A = \mathbb{Z}_q^* \times \mathbb{Z}_q^*$, and define

$$\mathcal{K} = \{(p, q, E, A, m, B) : B = mA\},$$

where $0 \leq m \leq q - 1$. The values $p, q, E, A$ and $B$ are the public key, and $m$ is the private key.

For $K = (p, q, E, A, m, B)$, and for a (secret) random number $k$, $1 \leq k \leq q - 1$, define

$$\text{sig}_K(x, k) = (r, s),$$

where

$$kA = (u, v)$$

$$r = u \mod q, \quad \text{and}$$

$$s = k^{-1}(\text{SHA-1}(x) + mr) \mod q.$$

(If either $r = 0$ or $s = 0$, a new random value of $k$ should be chosen.)

For $x \in \{0, 1\}^*$ and $r, s \in \mathbb{Z}_q^*$, verification is done by performing the following computations:

$$w = s^{-1} \mod q$$

$$i = w \text{SHA-1}(x) \mod q$$

$$j = wr \mod q$$

$$(u, v) = iA + jB$$

$$\text{ver}_K(x, (r, s)) = \text{true} \iff u \mod q = r.$$
Properties of Elliptic Curves

**Hasse bound**

\[ p + 1 - 2\sqrt{p} \leq \#E \leq p + 1 + 2\sqrt{p}. \]

\[ \uparrow \]

**Schoof algorithm**

\[ O(\log^3 p) - 64t \text{ ops} \]

**Theorem 6.1** Let \( E \) be an elliptic curve defined over \( \mathbb{Z}_p \), where \( p \) is prime and \( p > 3 \). Then there exist positive integers \( n_1 \) and \( n_2 \) such that \( (E, +) \) is isomorphic to \( \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \). Further, \( n_2 \mid n_1 \) and \( n_2 \mid (p - 1) \).

\( \text{cyclic subgroup of } E \text{ of order } 2^{160} \text{ is "safe" } \)

\( n_2 = 1 \text{ iff } E \text{ cyclic } \)
Implementations in Hardware and Software

- Elliptic curve computations usually regarded as consisting of four layers:
  - Basic modular arithmetic operations are computationally most expensive
  - Group operation implements point doubling and point addition
  - Point multiplication can be implemented using the Double-and-Add method
  - Upper layer protocols like ECDH and ECDSA

- Most efforts should go in optimizations of the modular arithmetic operations, such as
  - Modular addition and subtraction
  - Modular multiplication
  - Modular inversion
two NIST
Koblitz curves
in binary Galois fields

K163

\[ p(t) = t^{163} + t^7 + t^6 + t^3 + 1 \]
\[ a = 1 \]
\[ G_x = 2fe13c0537bcc11acaa07d793de4e6d5e5c94eee8 \]
\[ G_y = 289070fb05d38ff58321f2e800536d538ccdaa3d9 \]
\[ h = 5846006549323611672814741753598448348329118574063 \]
\[ h = 2 \]

K233

\[ p(t) = t^{233} + t^{74} + 1 \]
\[ a = 0 \]
\[ G_x = 17232ba853a7e731af129f22ff4149563a419c26bf50a4c9d6eefad6126 \]
\[ G_y = 1db537dece819b7f70f555a67c427a8cd9bf18aeb9b56e0c11056fae6a3 \]
\[ h = 3450873173395281893717377931138512760570940988862252126328087024741 \]
\[ h = 4 \]
SEC 2: Recommended Elliptic Curve Domain Parameters

Certicom Research
Contact: Daniel R. L. Brown (dbrown@certicom.com)

January 27, 2010
Version 2.0

©2010 Certicom Corp.
License to copy this document is granted provided it is identified as “Standards for Efficient Cryptography 2 (SEC 2)”, in all material mentioning or referencing it.
secp256k1 refers to the parameters of the ECDSA curve used in Bitcoin, and is defined in *Standards for Efficient Cryptography* (SEC) (Certicom Research, NIST).

This is a graph of secp256k1's elliptic curve \( y^2 = x^3 + 7 \) over the real numbers. Note that because secp256k1 is actually defined over the field \( \mathbb{Z}_p \), its graph will in reality look like random scattered points, not anything like this.

http://www.secg.org/sec2-v2.pdf)
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Section</th>
<th>Strength</th>
<th>Size</th>
<th>RSA/DSA</th>
<th>Koblitz or random</th>
</tr>
</thead>
<tbody>
<tr>
<td>secp192k1</td>
<td>2.2.1</td>
<td>96</td>
<td>192</td>
<td>1536</td>
<td>k</td>
</tr>
<tr>
<td>secp192r1</td>
<td>2.2.2</td>
<td>96</td>
<td>192</td>
<td>1536</td>
<td>r</td>
</tr>
<tr>
<td>secp224k1</td>
<td>2.3.1</td>
<td>112</td>
<td>224</td>
<td>2048</td>
<td>k</td>
</tr>
<tr>
<td>secp224r1</td>
<td>2.3.2</td>
<td>112</td>
<td>224</td>
<td>2048</td>
<td>r</td>
</tr>
<tr>
<td>secp256k1</td>
<td>2.4.1</td>
<td>128</td>
<td>256</td>
<td>3072</td>
<td>k</td>
</tr>
<tr>
<td>secp256r1</td>
<td>2.4.2</td>
<td>128</td>
<td>256</td>
<td>3072</td>
<td>r</td>
</tr>
<tr>
<td>secp384r1</td>
<td>2.5.1</td>
<td>192</td>
<td>384</td>
<td>7680</td>
<td>r</td>
</tr>
<tr>
<td>secp521r1</td>
<td>2.6.1</td>
<td>256</td>
<td>521</td>
<td>15360</td>
<td>r</td>
</tr>
</tbody>
</table>

Table 1: Properties of Recommended Elliptic Curve Domain Parameters over $\mathbb{F}_p$
2.4.1 Recommended Parameters secp256k1

The elliptic curve domain parameters over $\mathbb{F}_p$ associated with a Kobitz curve secp256k1 are specified by the sextuple $T = (p, a, b, G, n, h)$ where the finite field $\mathbb{F}_p$ is defined by:

\[
p = \text{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFC2F}
\]
\[= 2^{256} - 2^{32} - 2^{9} - 2^{8} - 2^{7} - 2^{6} - 2^{4} - 1
\]

The curve $E$: $y^2 = x^3 + ax + b$ over $\mathbb{F}_p$ is defined by:

\[
a = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
\]
\[
b = 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
\]

The base point $G$ in compressed form is:

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

and in uncompressed form is:

\[
G = 04 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8
\]

Finally the order $n$ of $G$ and the cofactor are:

\[
n = \text{FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF BAAEDCE6 AF48A03B BFD25E8C D0364141}
\]
\[
h = 01
My questions

1. What is current #0s in POW?  
   // around 70

2. Why not single SHA256?  
   // like HMAC

3. Domains for private and public keys are close but not the same.  
   uncompressed private keys make no sense
   \[ p \text{ and } |<G>| \text{ are distinct and have different roles} \]

4. Is nonce k in ElGamal "child private key"?

5. Does say
   \[ \text{private_key} = \text{SHA256(mini_key)} \]
   see 3 above, looks incorrect
<table>
<thead>
<tr>
<th>Year</th>
<th>BTC</th>
<th>Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>2012</td>
<td>10M</td>
<td>25</td>
</tr>
<tr>
<td>2017</td>
<td>17M</td>
<td>12.5</td>
</tr>
<tr>
<td>2018</td>
<td></td>
<td>6.25</td>
</tr>
</tbody>
</table>

Soon

Over 21M

Limit

10 Ex Ha per second

$10^{19} \approx 2^{64}$

Mining revolution
One-time Signatures

Winternitz OTS

used in IOTA

Cryptosystem 7.6: Lamport Signature Scheme

Let $k$ be a positive integer and let $\mathbb{P} = \{0, 1\}^k$. Suppose $f : Y \to Z$ is a one-way function, and let $A = Y^k$. Let $y_{i,j} \in Y$ be chosen at random, $1 \leq i \leq k$, $j = 0, 1$, and let $z_{i,j} = f(y_{i,j})$, $1 \leq i \leq k$, $j = 0, 1$. The key $K$ consists of the $2k$ $y$’s and the $2k$ $z$’s. The $y$’s are the private key while the $z$’s are the public key.

For $K = (y_{i,j}, z_{i,j} : 1 \leq i \leq k, j = 0, 1)$, define

$$\text{sig}_K(x_1, \ldots, x_k) = (y_1, x_1, \ldots, y_k, x_k).$$

A signature $(a_1, \ldots, a_k)$ on the message $(x_1, \ldots, x_k)$ is verified as follows:

$$\text{ver}_K((x_1, \ldots, x_k), (a_1, \ldots, a_k)) = \text{true} \iff f(a_i) = z_{i,x_i}, 1 \leq i \leq k.$$
Example 7.6  7879 is prime and 3 is a primitive element in \( \mathbb{Z}_{7879}^* \). Define

\[
f(x) = 3^x \mod 7879.
\]

Suppose \( k = 3 \), and Alice chooses the six (secret) random numbers

\[
\begin{align*}
y_{1,0} &= 5831 \\
y_{1,1} &= 735 \\
y_{2,0} &= 803 \\
y_{2,1} &= 2467 \\
y_{3,0} &= 4285 \\
y_{3,1} &= 6449.
\end{align*}
\]

These \( z \)'s are published. Now, suppose Alice wants to sign the message

\[
x = (1, 1, 0).
\]

The signature for \( x \) is

\[
(y_{1,1}, y_{2,1}, y_{3,0}) = (735, 2467, 4285).
\]

To verify this signature, it suffices to compute the following:

\[
\begin{align*}
3^{735} \mod 7879 &= 3810 \\
3^{2467} \mod 7879 &= 4721 \\
3^{4285} \mod 7879 &= 268.
\end{align*}
\]

Hence, the signature is verified.