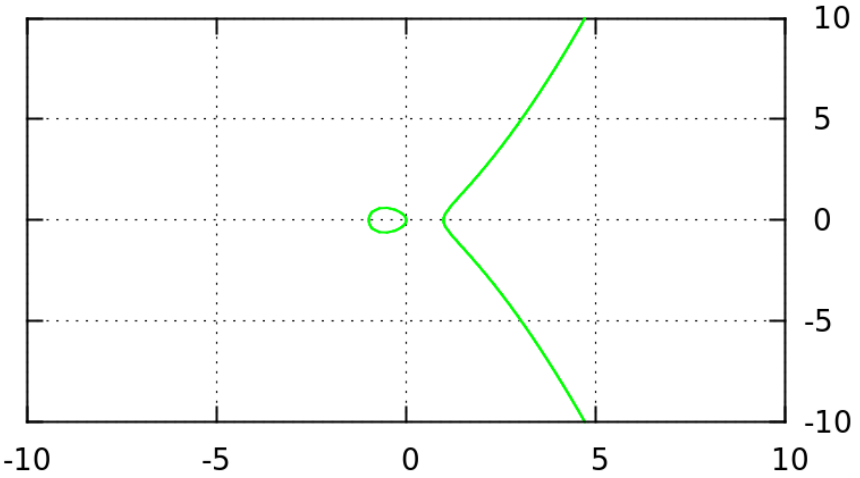
Kryptografie mittels elliptischer Kurven



Thema	*Kryptografie mittels elliptischer Kurven*
Vortragender	*Maurice Tollmien*
Studiengang	*Master Informatik*
Matrikelnummer	*inf101074*
Schriftliche Ausarbeitung	github.com/MauriceGit/Elliptic_Curve_Cryptography_Seminar

Inhalt

- Kurzer Einblick in die Mathematik
- Das Diskrete Logarithmus Problem
- Allgemeine Funktionsweise
- Asynchroner Schlüsseltausch
  - Diffie-Hellmann
- Signieren
- Angriffe und Zeitkomplexität
- Schlüssellängen

Kurzer Einblick in die Mathematik

Berechnungen im Kontext einer algebraischen Gruppe G.

Generator α, für den gilt, dass $\forall a \in G \; \exists i \in \mathbb{N} : \alpha^i = \alpha \cdot \alpha \cdot \cdots \alpha = a$.

Die Ordnung eines Elementes $\alpha \in G$ ist die niedrigste Zahl n für die gilt: $\alpha^n = e \; mod \; n$
mit $e =$ dem neutralen Element bezüglich des Operators $\cdot$ in G.

Das Diskrete Logarithmus Problem

Wie wir es bisher kannten:

Sei G eine zyklische Gruppe und $g \in G$ ein erzeugendes Element.

Der kleinste Wert m, welches $h = g^m$ erfüllt, ist der sogenannte Logarithmus von h im Bezug auf g. Also: $m = log_g(h)$.

Auf elliptischen Kurven:

Gegeben seien Punkte P und Q auf einer elliptischen Kurve.

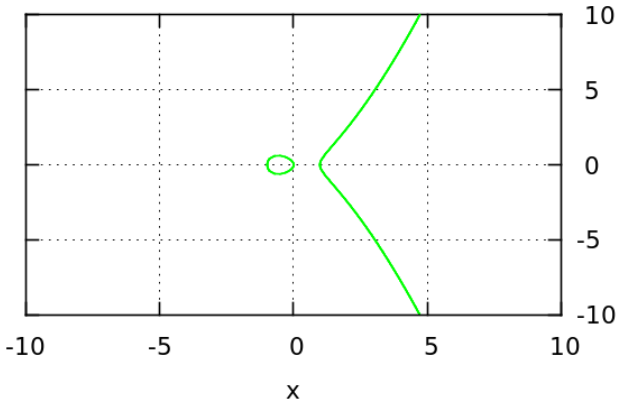
Folgende Gleichung muss gelöst werden: $P * k = Q$. k ist wird der diskrete Logarithmus von Q auf Basis von P genannt.

Allgemeine Funktionsweise

$$y^2 = x^3 + a * x + b$$

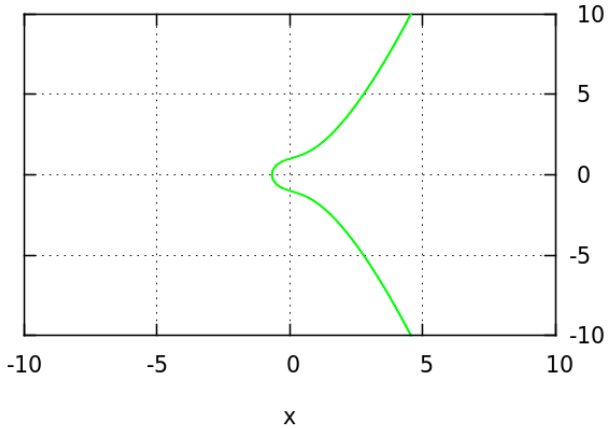
Eine elliptische Kurve definiert sich über seine beiden freien Parameter a und b.

$$y^2 = x^3 - x$$



$$E(-1, 0)$$

$$y^2 = x^3 + x + 1$$



$$E(1, 1)$$

Allgemeine Funktionsweise

Jede Gerade, definiert durch zwei Punkte auf der Kurve, besitzt genau einen dritten Schnittpunkt mit der Kurve.

P und Q seien Punkte auf der elliptischen Kurve.

O ist ein Punkt auf der Kurve, welcher per Konvention im Unendlichen liegt und das neutrale Element darstellt.

Die Negation eines Punktes $P = (x, y)$ ist definiert als $-P = (x, -y)$.

Die Addition von zwei Punkten P und Q ist die Negation des dritten Schnittpunktes der Geraden zwischen P und Q.

Die skalare Multiplikation eines Punktes P ist definiert als wiederholte Addition.

Asynchroner Schlüsseltausch

Die folgenden Attribute gelten für alle vorzustellenden Algorithmen:

Als E wird die elliptische Kurve bezeichnet aus $GF(q)$ wobei $q = p^n$ eine bevorzugt große Zahl darstellt.

Es muss eine öffentlich zugängliche Funktion existieren, welche eine Nachricht m auf einen Punkt P_m der elliptischen Kurve abbildet: $f : m \rightarrow P_m$.

Die Kurvenparameter der genutzten Kurve sind öffentlich.

Einige der uns bekannten Algorithmen, wie Diffie-Hellman, Massey-Omura oder ElGamal sind auch auf elliptische Kurven anwendbar.

Diffie-Hellmann

- Erzeugendes Element G ist öffentlich
- Alice und Bob erzeugen ein Schlüsselpaar (d, Q) mit einem privaten Schlüssel d mit $0 < d < N$ und einem öffentlichen Schlüssel $Q = d * G$.
- Alice berechnet nun $d_A * Q_B = (x_k, y_k)$. Bob berechnet $d_B * Q_A = (x_k, y_k)$.
- Die x-Koordinate x_k ist nun das gemeinsame Geheimnis.
- Das funktioniert, weil:
  $d_A * Q_B = d_A * d_B * G = d_B * d_A * G = d_B * Q_A$.

Signieren

- $e = HASH(m)$
  Alice erzeugt einen kryptografisch sicheren Hash aus der Nachricht m.
- z seien die n linkesten Bits aus e, mit n gleich der Bitlänge der Gruppenordnung.
- Alice berechnet einen Zufallswert k aus dem Interval $[1, n-1]$.
- $(x_1, y_1) = k * G$
  Alice berechnet den Kurvenpunkt $k * G$.
- $r = x_1$
  $s = k^{-1}(z + r * d_A)$
- (r, c) ist die Signatur der Nachricht m.

Angriffe und Zeitkomplexität

Zum aktuellen Zeitpunkt sind keine theoretischen oder praktischen Angriffe auf Kurven mit gut gewählten Parametern bekannt.

Weiterhin sind keine Algorithmen bekannt, die das Problem in subexponentieller Zeit lösen.

Mit den Pollard-λ und Pollard-ρ-Methoden lassen sich jedoch Kurven in:

Pollard-ρ: $\sqrt{\pi * n/2}$ Schritten

und

Pollard-λ: $2 * \sqrt{n}$ Schritten

berechnen.

Schlüssellängen

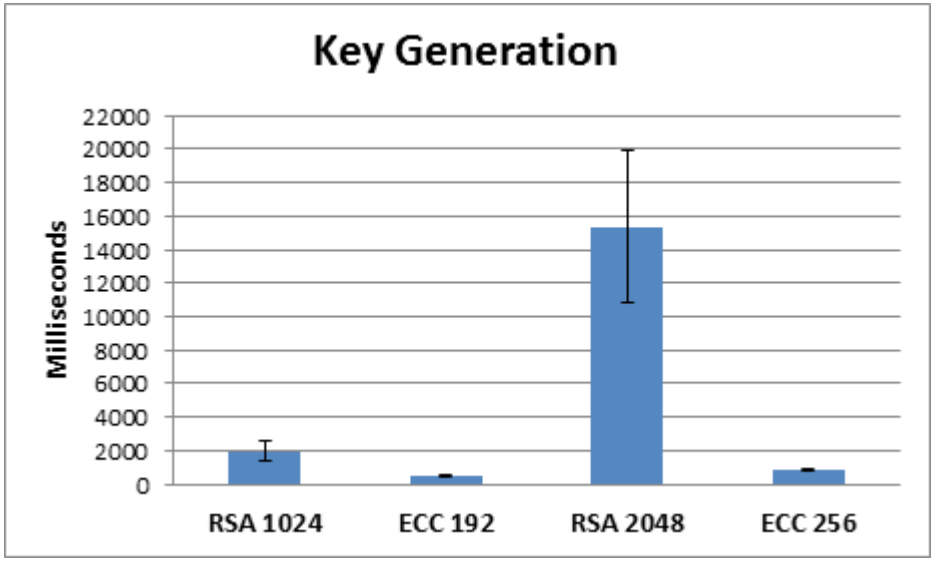
Schlüssellänge in bits	Pollard-ρ	MIPS years
160	2^{80}	$8.5 * 10^{11}$
192	2^{96}	$5.6 * 10^{16}$
224	2^{112}	$3.7 * 10^{21}$
256	2^{128}	$2.4 * 10^{26}$
384	2^{192}	$4.4 * 10^{45}$
521	2^{260}	$1.3 * 10^{66}$

Schlüssellängen

Sicherheits-bits	Symmetrische Algorithmen	FFC[5] zB. DSA, D-H	IFC[6] zB. RSA	ECC[7] zB. ECDSA
80	2TDEA	$L = 1024$ $N = 160$	$k = 1024$	$f = 160 - 223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224 - 255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256 - 283$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 284 - 511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

[5] Finite field cryptography
[6] Integer-factorization cryptography
[7] Elliptic curve cryptography

Schlüssellängen



Jedoch:

- Der Schlüsselaustausch mit RSA ist signifikant schneller, als mit ECC.
- Das Signieren mit RSA ist signifikant schneller, als mit ECC.
- Die Arbeitsspeicherauslastung ist ungefähr gleich bei RSA und ECC.

[1] http://people.cs.uct.ac.za/ spetzer/mobileMedRecords/results_shelley.html

Ende

Ein paar Codebeispiele und ein bisschen Spielen

Fragen und Anmerkungen ?