Fortgeschrittene Techniken der Kryptographie

Jürgen Fuß

Episode 10: Effizienz elliptischer Kurven

HAGENBERG | LINZ | STEYR | WELS

FH OBERÖSTERREICH
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

Vergleich von Schlüssellängen

Aufwand	Blockchiffre	MAC	Hashfunktion	RSA/DH	EC
2^{128}	128 (AES-128)	128 (AES-128-CMAC)	256 (SHA-256)	3072	256
2^{192}	192 (AES-192)	192	384	7680	384
2^{256}	256 (AES-256)	256 (SHA-256-HMAC)	512 (SHA-512)	15360	512

Beispiel: Sicherheitsniveau 128 Bit (1)

$\mathbb{Z}_p^*$ $p \approx 3072$ Bit (Index Calculus), $\omega \approx 256$ Bit.

EC mod p Ordnung der Kurve ≈ 256 Bit (BSGS).

Nach dem Satz von Hasse bedeutet dies, dass eine elliptische Kurve modulo p mit einer 256 Bit langen Primzahl p verwendet werden kann. $\omega \approx 256$ Bit wie zuvor.

Beispiel: Sicherheitsniveau 128 Bit (2)

$\mathbb{Z}_p^*$	Potenzieren von 3072 Bit langen Zahlen mit 256 Bit langen Exponenten.
EC mod p	Punktmultiplikation mit 256 Bit langen Faktoren, rechnen mit 256 Bit langen Zahlen.

- Eine Punktaddition entspricht in etwa 15 Multiplikationen (Kehrwert ≈ 10 Multiplikationen).
- In $\mathbb{Z}_p^*$ wird aber mit Zahlen gerechnet, die 12-mal so lang sind, Multiplikationen sind daher $12 \cdot 12 = 144$ Mal so aufwendig.
- Insgesamt ist der Aufwand so in $\mathbb{Z}_p^*$ ca. 10-mal so hoch.

Weitere Performancesteigerung

Punktkompression

- Punkte auf elliptischen Kurven können für die Übertragung oder Speicherung komprimiert werden.
- Ist $P = (x, y)$ ein Punkt auf einer elliptischen Kurve $\mathcal{E}$ über $\mathbb{Z}_p$, dann ist auch $-P = (x, -y) \in \mathcal{E}$. Dies sind die einzigen Punkte auf $\mathcal{E}$ mit x-Koordinate x.
- Ist x bekannt, lässt sich $\pm y$ bestimmen ($y = \pm\sqrt{x^3 + ax + b}$). $y^2 = x^3 + ax + b$
  Welche der beiden Lösungen $\pm y$ die richtige y-Koordinate ist, muss noch gespeichert werden.
- Ist $y < p/2$, dann ist $-y = p - y > p/2$, und umgekehrt. Es reicht also, die Information zu speichern, ob $y < p/2$. Dafür reicht ein Bit.

Signed Double-&-Add

$62 = 111110$

- Klassisches Double & Add

$$62 \cdot P = 32 \cdot P + 16 \cdot P + 8 \cdot P + 4 \cdot P + 2 \cdot P$$

braucht 5 Verdopplungen und 4 Additionen.

- Signed Double & Add

$$62 \cdot P = 64 \cdot P - 2 \cdot P$$

braucht 6 Verdopplungen, 1 Addition und 1 Inversion $(-)$. Die Berechnung des Inversen ist hier aber kein Aufwand $(-(x, y) = (x, -y))$.

Projektive Koordinaten

$\frac{3}{2} = \frac{6}{4} = \frac{12}{8} = \frac{21}{14}$ $\quad\quad$ $\frac{7}{12} + \frac{3}{18} = \frac{7 \cdot 18 + 3 \cdot 12}{12 \cdot 18}$

Den meisten Aufwand bei Punktadditionen machen die Divisionen. Eine Möglichkeit, diese Divisionen (weitestgehend) zu vermeiden, ist die Verwendung von **projektiven Koordinaten**. Dabei werden Punkte nicht in der Form (x, y) gespeichert, sondern in der Form $(X : Y : Z)$.

Ein Punkt $(X : Y : Z)$ mit $Z \neq 0$ steht dann stellvertretend für den Punkt $(X/Z, Y/Z)$. Die Koordinaten werden also sozusagen als Brüche (mit dem gleichen Nenner Z) gespeichert.

Umgekehrt ist die Darstellung eines Punktes (x, y) in projektiven Koordinaten $(X : Y : Z)$ nicht eindeutig, denn $(\lambda X : \lambda Y : \lambda Z)$ ist ebenfalls Repräsentant für (x, y) für jedes $\lambda \neq 0$.

Der Punkt ∞ wird in projektiven Koordinaten als $(0 : Y : 0)$ mit beliebigem $Y \neq 0$, am einfachsten als $(0 : 1 : 0)$, dargestellt.

$(\frac{1}{0}, \frac{1}{0})$

$\frac{0}{0} = 3 \quad 0 = 3 \cdot 0$ ✓

$\frac{0}{0} = 5 \quad 0 = 5 \cdot 0$ ✓

Umwandeln in projektive Koordinaten und umgekehrt

Das Umwandeln von normalen Koordinaten[1] in projektive ist einfach:

$$(x, y) \to (x : y : 1)$$
$$\infty \to (0 : 1 : 0).$$

Umgekehrt ist die Berechnung eines Kehrwerts erforderlich:

$$(X : Y : Z) \to (XZ^{-1}, YZ^{-1})$$
$$(0 : Y : 0) \to \infty.$$

[1] Normale Koordinaten werden in der Literatur oft auch als affine Koordinaten bezeichnet.

Rechnen mit projektiven Koordinaten

Für die Summe zweier Punkte $P := (X_1 : Y_1 : Z_1)$ und $Q := (X_2 : Y_2 : Z_2)$ in projektiven Koordinaten erhält man Formeln, die ohne aufwendige Divisionen auskommen.

$$
\begin{aligned}
u &:= Y_2 Z_1 - Y_1 Z_2, \\
v &:= X_2 Z_1 - X_1 Z_2, \\
w &:= u^2 Z_1 Z_2 - v^3 - 2v^2 X_1 Z_2, \\
P + Q &:= \Big(vw \ : \ u(v^2 X_1 Z_2 - w) - v^3 Y_1 Z_2 \ : \ v^3 Z_1 Z_2 \Big).
\end{aligned}
$$

Damit lassen sich Punkte mit zwölf Multiplikationen und zweimaligem Quadrieren, aber ohne eine Division, addieren.
Für die Punktverdopplung lassen sich analog Formeln bestimmen.

Vorteil von projektiven Koordinaten

Bei einer Punktmultiplikation kann während des Double-&-Add-Prozesses durchgehend in projektiven Koordinaten gerechnet werden, eine Kehrwertberechnung ist dann erst ganz am Ende einmal für das Umrechnen in affine Koordinaten erforderlich, aber nicht für jede Punktaddition/-verdopplung.

Montgomery Curves

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

Montgomery Curves haben eine etwas andere Kurvengleichung:

$$By^2 = x^3 + Ax^2 + x \pmod p$$

Formeln für die Addition von Punkten lassen sich analog zu vorher herleiten. Speziell für die Punktmultiplikation ergeben sich unter Verwendung von projektiven Koordinaten sehr effiziente Formeln. Insbesondere können in diesen Formeln die y-Koordinaten der Punkte ignoriert werden.

Punktmultiplikation auf Montgomery Curves

Sind $(X_1 : \_ : Z_1)$ die projektiven Koordinaten von P, $(X_n : \_ : Z_n)$ die projektiven Koordinaten des Punkts $n \cdot P$ und $(X_{n+1} : \_ : Z_{n+1})$ die projektiven Koordinaten des Punkts $(n+1) \cdot P$, dann lassen sich die projektiven Koordinaten $(X_{2n+1} : \_ : Z_{2n+1})$ und $(X_{2n} : \_ : Z_{2n})$ der Punkte $(2n+1) \cdot P$ und $(2n) \cdot P$ daraus wie folgt berechnen:

$$X_{2n+1} = Z_1((X_{n+1} - Z_{n+1})(X_n + Z_n) + (X_{n+1} + Z_{n+1})(X_n - Z_n))^2,$$
$$Z_{2n+1} = X_1((X_{n+1} - Z_{n+1})(X_n + Z_n) - (X_{n+1} + Z_{n+1})(X_n - Z_n))^2.$$

$$X_{2n} = p_n \cdot q_n,$$
$$Z_{2n} = (p_n - q_n)(q_n + \frac{A+2}{4}(p_n - q_n)), \text{ wobei}$$
$$p_n = (X_n + Z_n)^2 \text{ und } q_n = (X_n - Z_n)^2.$$

Nutzt man zum Berechnen von Vielfachen eines Punktes die unter dem Namen **„Montgomery Ladder“** bekannte Methode, so werden stets nur die beiden angegebenen Formeln benötigt.

Montgomery Ladder

Gegeben: G, $\alpha = (b_0, b_1, \ldots, b_m)_2$
Gesucht: $A := \alpha \cdot G$

$A \leftarrow \infty$
$B \leftarrow G$
for $i = 0, \ldots, m$
    if $b_i = 0$ then
        $B \leftarrow A + B$
        $A \leftarrow 2A$
    else
        $A \leftarrow A + B$
        $B \leftarrow 2B$
return A

Wesentlich für die Anwendung bei Montgomery Curves ist, dass zu jedem Zeitpunkt in diesem Algorithmus sowohl A als auch B Vielfache des Punktes G sind. Genauer: Stets gibt es eine nichtnegative ganze Zahl n, so dass $A = n \cdot G$ und $B = (n+1) \cdot G$.
Am Beginn ist dies klar ($n = 0$). Ist zu irgendeinem Zeitpunkt $A = n \cdot G$ und $B = (n+1) \cdot G$, dann betrachten wir die Änderungen in den beiden if-Zweigen.
Im ersten if-Zweig wird

- A zu $2 \cdot A = 2n \cdot G$ und
- B zu $A + B = n \cdot G + (n+1) \cdot G = (2n+1) \cdot G$.

Im zweiten if-Zweig

- A zu $A + B = n \cdot G + (n+1) \cdot G = (2n+1) \cdot G$ und
- B zu $2 \cdot B = 2(n+1) \cdot G = (2n+2) \cdot G$.

In beiden Fällen sind A und B wieder Vielfache von G und der Koeffizient vor B ist um 1 größer als jener von A.

Curve25519 ist eine Montgomery-Kurve, für welche die Berechnungen besonders effizient implementiert werden können. Dafür werden als Parameter

$$\begin{aligned} p &= 2^{255} - 19 \qquad \text{(daher der Name)} \\ A &= 486662 \qquad \text{und} \\ B &= 1 \end{aligned}$$

$2^{255} - 19 = 0 \mod p$
$2^{255} = 19$
$2^{256} = 38 \mod p$

gewählt. Als Basispunkt für ECDH dient der Punkt $G = (9 : \_ : 1)$ mit der primen Ordnung

$$\omega = 2^{252} + 27742317777372353535851937790883648493. \in \mathbb{P}$$

Leider ist die Ordnung der Kurve nicht ω, sondern $8 \cdot \omega$. Es gibt auf dieser Kurve also auch Punkte der Ordnung 2, 4 und 8, die in Bezug auf das DLP problematisch sind. In Protokollen ist es daher ggf. erforderlich, zu testen, ob solche Punkte auftauchen.

Ed25519

Für besonders effiziente Signaturen mit elliptischen Kurven wird gerne das Verfahren Ed25519 eingesetzt, das hier ganz kurz dargestellt werden soll.
Als Gruppe für die Berechnungen wird hier die elliptische Kurve mit der Gleichung

$$-x^2 + y^2 = 1 + dx^2y^2 \pmod{p},$$

mit den Parametern $p = 2^{255} - 19$ (daher der Name) und $d = -\frac{121665}{121666}$ verwendet.
Diese Kurve, eine sogenannte Edwards-Kurve, ist eng verwandt mit Curve25519.
Der Punkt mit y-Koordinate $4/5 \pmod{p}$ hat die prime Ordnung

$$\omega = 2^{252} + 27742317777372353535851937790883648493.$$

Die Ordnung der Kurve ist 8ω.

$r = (g^x \cdot A^y \bmod p) \bmod \omega$

$r = (x \cdot G + y \cdot A)_x \bmod \omega$

Rechnen auf Ed25519

Es sei $p = 2^{255} - 19$ und $d = -\frac{121665}{121666}$. Weiterhin sei

$$\mathcal{E} := \left\{ (x, y) \in (\mathbb{Z}_p)^2 \,\middle|\, -x^2 + y^2 = 1 + dx^2y^2 \right\}.$$

Man definiert man für $P = (x_1, y_1) \in \mathcal{E}$ und $Q = (x_2, y_2) \in \mathcal{E}$

$$\begin{aligned}
\perp &:= (0, 1), \\
-P &:= (-x_1, y_1) \text{ und} \\
P + Q &:= (x_3, y_3), \text{ mit} \\
t &:= dx_1x_2y_1y_2, \\
x_3 &:= \frac{x_1y_2 + x_2y_1}{1 + t} \qquad \text{und} \\
y_3 &:= \frac{y_1y_2 + x_1x_2}{1 - t},
\end{aligned}$$

Ed25519 vs. ECDSA (1)

Keine zufälligen Parameter. Für ECDSA-Signaturen wird ein zufälliger Wert k gewählt, um

$$r := (k \cdot G)_x \bmod \omega$$

zu berechnen. Als Schutz vor Nonce Reuse wird Ed25519 deshalb k als Hashwert über einen Teil des Private Key und die zu signierende Nachricht berechnet. Wird die selbe Nachricht mit dem selben Private Key ein zweites Mal signiert, so ergibt sich exakt die gleiche Signatur.

Ed25519 vs. ECDSA (2)

Hashwert hängt nicht nur von der Nachricht ab. ECDSA-Signaturen berechnen den zweiten Teil der Signatur nach der Formel

$$s := k^{-1}(H(m) + \alpha r) \bmod \omega.$$

Bei Ed25519 hängt der Hashwert nicht allein von m ab, sondern zusätzlich vom Public Key und dem zuvor berechneten Wert r. Durch diese Veränderung reicht ein Brechen der (starken) Kollisionsresistenz der Hashfunktion nicht mehr, um Signaturen zu fälschen. Diese Konstruktion ist dem Schnorr-Signaturverfahren „abgeschaut“. Schließlich wird der Wert s ebenfalls ähnlich wie bei Schnorr-Signaturen als

$$s := k + \alpha H(r, A, m) \mod \omega$$

berechnet. Die Prüfung einer Signatur erfolgt dann auch analog zu Schnorr-Signaturen.