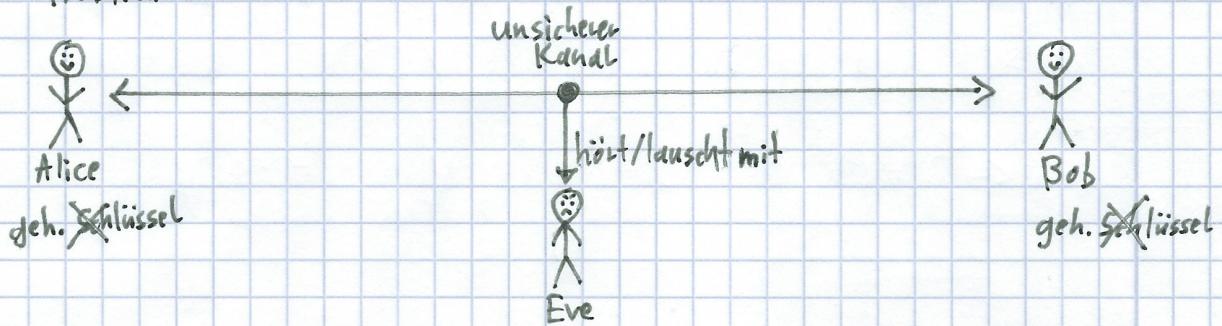


Eigenes „Gesamtbild“ CRYPTO1 – „Überblick & Einführung“

14.5.22

Problem:



Unlösbar?! Ralph Merkle sagte 1974/78: Nein!
↳ „Secure Communications over Insecure Channels“

Vorschlag 1: Schlüssel raten	(benötigt Einwegfunktion f)	
Alice:	Bob:	Eve:
d_1	$f(a_1) \rightarrow$	Eve hat: $f(a_1)$
$\leftarrow f(a_2)$	d_2	$f(a_2)$
d_3	$f(a_3) \rightarrow d_3$	$f(a_3)$
$\leftarrow f(a_4)$	d_4	\vdots
d_5	$f(a_5) \rightarrow d_5$	und $f(a_{2j+1}) = f(a_{2i})$, kennt aber nicht $a_{2j+1} = a_{2i}$. Dieses muss sie über Brute-Force heraus- finden und hat dabei Aufwand $O(N)$.
\vdots	\ddots	
$\therefore f(a_{2j+1}) = f(a_{2i}) \Rightarrow d_{2j+1} = a_{2i}$		
Wenn $a_k \in [1, N]$, dann tritt nach $O(\sqrt{N})$ Schritten eine Kollision auf (Geburtstagsparadoxon). Der Aufwand von Alice & Bob beträgt also $O(\sqrt{N})$.		

Vorschlag 2: Merkle-Puzzle	(Rätsel)	
Alice:	Bob:	Eve:
Erstellt eine Menge von Rätseln und schickt diese an Bob: $R_1, R_2, R_3, R_4, R_5, \dots, R_N$ Die Schwierigkeit eines Rätsels ist $O(N)$.	Wählt R_i und löst es: K_i	Eve hat: R_1, R_2, R_3, \dots
	$f(K_i) \leftarrow$ gemeinsamer Schlüssel	und $f(K_i)$, kennt aber weder K_i noch i. Dieses muss sie über Brute-Force durch Lösen aller (im Schnitt der Hälfte aller) Rätsel herausfinden. Aufwand: $O(N^2)$
Aufwand Alice: $O(N)$ zum Erstellen von N Rätseln.	Aufwand Bob: $O(N)$ zum Lösen eines Rätsels.	

Wichtige Details:

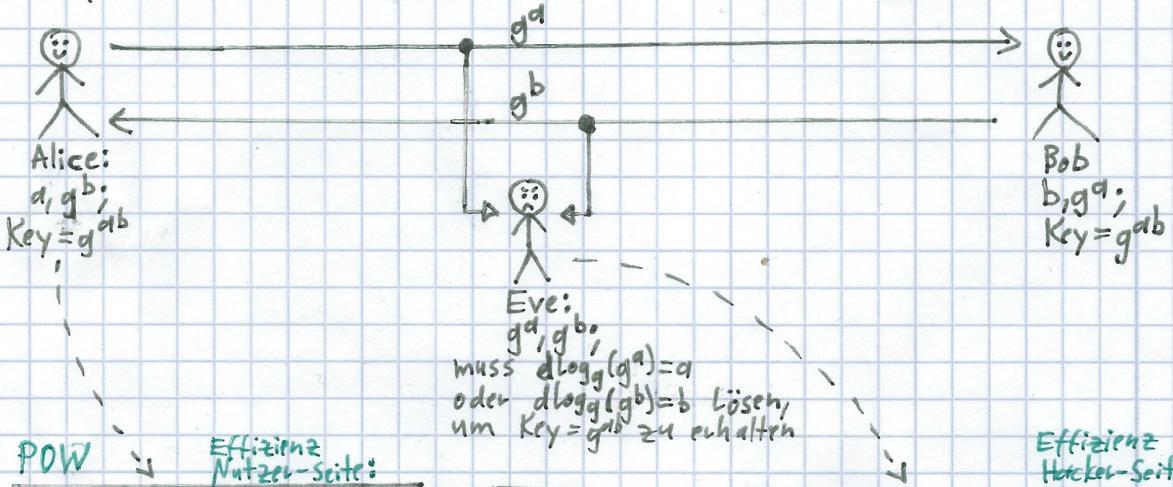
- Eine Einwegfunktion f ist nicht zwingend notwendig. Die Rätsel können auch in zufälliger Reihenfolge von Alice verschoben und der Index mit verschlüsselt werden. Dann kann Bob den Index i im Klartext zurücksenden.
- Die Chiffre müssen Redundanz enthalten, damit Bob sie erkennt.

BRONNEN

Eigenes „Gesamt Bild“ CRYPTO 1 – Diskr. Log. (I) + 5.22 Diffie-Hellman-Schlüsselaustausch zw. Alice & Bob: + 5.22 & 7.5.22

15/16.5.22

Diffie-Hellman-Schlüsselaustausch zw. Alice & Bob:

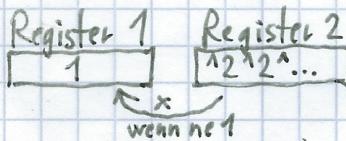


Wie berechnet Alice g^a und $(g^b)^a$ effizient?:

Square and Multiply:

$$g^1 \xrightarrow{\text{square}} g^{10} \xrightarrow{\text{square}} g^{100} \rightarrow \dots$$

$$\begin{aligned} \text{z.B. } & g^{18} \\ &= g^{100010} \\ &= g^{100000} \cdot g^{10} \end{aligned}$$



$\Rightarrow O(\log(g^a))$ Platz 2
 $\Rightarrow O(\log(a))$ Zeit

(VL Diskr. Log. II): Weiterhin „knacke“ $d\log$

1.b.) Indexkalkül / Index Calculus [1.] wähle ein B.

[2.] Bestimme Zahlen f_1, \dots, f_r sodass

gfü B-glatt (d.h. alle Primteiler $\leq B$).

Faktorbasis $f_B = \{p_1, \dots, p_t\} = \{p \mid p \in P, p \leq B\}$.

$g^f_i = p_1^{e_1} \cdots p_t^{e_t}$

$\Rightarrow f_i = e_1 \cdot d\log(p_1) + \cdots + e_t \cdot d\log(p_t)$

\rightarrow LGS mit r Gleichungen & t Unbekannten

\rightarrow Lösung liefert $d\log(p_i)$, $1 \leq i \leq t$.

\rightarrow LGS ist mod $p-1$ lösbar $\Leftrightarrow p-1 \in P$, sonst chot.

[3.] Suche ein f_i sodass $h \cdot g^{f_i}$ B-glatt:

$$h \cdot g^f = p_1^{e_1} \cdots p_t^{e_t} \Rightarrow d\log(h) + f = e_1 \cdot d\log(p_1) + \cdots + e_t \cdot d\log(p_t)$$

$$\Rightarrow d\log(h) = -f + \sum_{i=1}^t e_i \cdot d\log(p_i)$$

(VL Diskr. Log. II): schneller Θ fb finden nur für 2^p machbar, n. Ell.K.

Spare Platz bei Pollard-Rho mit „Floyd's Trick“ (dafür etwas mehr Zeit nötig):

$$x_0 = y_0 = g^{a_0} \cdot h^{b_0}; \text{ merke } x_i, y_i \text{ "Durstig"}$$

$$x_i := f(x_{i-1}), y_i := f(f(y_{i-1}))$$

$$\text{Sobald } x_k = y_k \Leftrightarrow g^{ak} \cdot h^{bk} = g^{ak} \cdot h^{bk}$$

$$\Rightarrow g^{ak-bk} = h^{bk-bk} = (g^b)^{ak-bk}$$

$$\Rightarrow x = g^{ak-bk}$$

$$\Rightarrow x = g^{ak-bk} \quad \text{Eine Kollision existiert}$$

$$\Rightarrow x = g^{ak-bk} \quad \text{bei } y_i = x_i \Leftrightarrow i \equiv 0 \pmod{t}$$

Wie berechnet Eve $x = d\log(g^X)$, gegeben $h = g^X$, effizient?:

1.) Silver-Pohlig-Hellman-Reduktion: $G = \langle g^p, \dots, g^{p-1} \rangle$ zyklisch.

[1] $\text{ord}(G) = a \cdot b$ mit $\text{ggT}(a, b) = 1$ und DLP ist „leicht“ in Ordnung a und b :

• Finde „leicht“ x_a, x_b : $(g^a)^{x_a} = h^a$ (Ord. a)

• Finde mit Euklid u, v : $ua + vb = \text{ggT}(a, b) = 1$

• Ergebnis: $x = u \cdot a \cdot x_a + v \cdot b \cdot x_b$

• Beweis: $g^x = (g^{ax_b})^u \cdot (g^{bx_a})^v = h^{au+bv} = h$

[2] $\text{ord}(G) = p^e$ mit $p \in P$ und $e > 1$ und DLP ist „leicht“ in Ordnung p^e :

Idee: Reduziere rekursiv von Ord. p^e auf p :

• $x = g^x$

• $h = g^{x_0+x_1p+x_2p^2+\dots}$ (schreibe x zur Basis p)

• $h^{p^{e-1}} = (g^{x_0+x_1p+\dots})^{p^{e-1}}$

• $h^{p^{e-1}} = (g^{p^{e-1}})^{x_0} \cdot g^{x_1+\dots+p^{e-1} \cdot (x_1+x_2p+\dots)}$

• $h^{p^{e-1}} = (g^{p^{e-1}})^{x_0} \cdot g^1$ (da Ordnung p^e)

• $\text{ord}(\langle g^{p^{e-1}} \rangle) = p$, also x_0, x_1, \dots bestimbar

• Neues Problem: $h = g^{x_0} \cdot (g^p)^{x_1+x_2p+\dots}$

$h \cdot g^{-x_0} = (g^p)^{x_1+x_2p+\dots}$

\rightarrow DLP in $\langle g^p \rangle$ der Ordnung p^{e-1} (zyklisch)

\Rightarrow Bleibt das DLP in Gruppen mit Primzahlordnung:

2.a.) Baby step-Giant step-Algorithmus / Shanks' Algo: (beliebige $(kg)^l = n$) zykl. Grp.

Giant Steps: $g^0, g^N, g^{2N}, \dots, g^{(N-1)N}$ ($N = \lceil \sqrt{n} \rceil$)

Baby Steps: h, hg, hg^2, \dots

Irgendwann muss Kollision: $g^{uN} = hg^{-v}$

$\Rightarrow x = d\log_g(h) = uN + v$

2.b.) Pollard-Rho („Rho“): $f(x) = \frac{x_h - x_E}{x_q - x_G}$ $\Rightarrow x_i = g^{u_i} \cdot h^{v_i}$

$x_L = x_L + f \Rightarrow x_L = x_L + \frac{x_L - x_{L-1}}{x_{L-1} - x_{L-2}} \Rightarrow x_L = x_L + \frac{x_L - x_{L-1}}{x_{L-1} - x_{L-2}} + \dots$

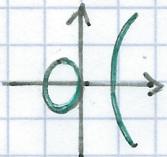
$\Rightarrow g^{u_L-v_L} = h^{v_L-u_{L-1}} = (g^x)^{v_L-u_{L-1}-v_L} = (g^x)^{u_L-u_{L-1}-v_L} \Rightarrow x = \frac{u_L-u_{L-1}}{u_L-u_{L-1}-v_L}$

\Rightarrow Beide basieren auf dem Geburtsdaysparadoxon ($\sqrt{|G|}$) und sind besser als naives Brute-Force, aber dennoch für große Gruppen n. durchf. bzw.

Eigenes „Gesamtbild“ CRYPTO1 – Ell. Kurven (0+I)

18/20/21/22.5.12

$E_{a,b} = \{(x,y) \in F^2 \mid y^2 = x^3 + ax + b\} \cup \{0\}$ ist EC über Körper F , wobei $2^2 \cdot a^3 + 3^3 \cdot b^2 \neq 0$ und für die Vorlesung $F = \mathbb{Z}_p$, $p \in \mathbb{P}$, $p \geq 5$.
(z.B. $a=-3, b=2$)
 $E_{1,0}$ bzw. $y^2 = x^3 - x$: $E_{1,1}$ bzw. $y^2 = x^3 - x + 1$:

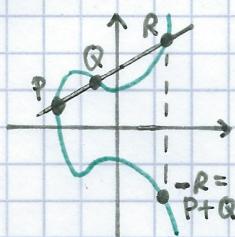


$$\text{Definition der Gruppenoperation/Addition: } (x_1, y_1) + (x_2, y_2) = (x_3, y_3) :=$$

Fall 0:

$$\begin{aligned} & P + 0 \\ &= 0 + P \\ &= P \end{aligned}$$

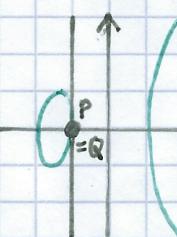
Fall 1:



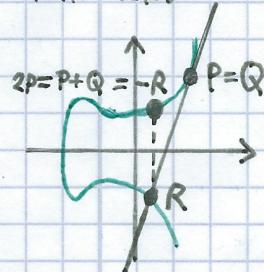
Fall 2(a):



Fall 2(b):

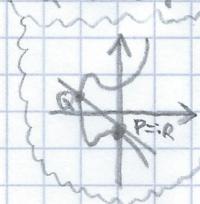


Fall 2(c):



$(x_1 = x_2, y_1 \neq y_2, y_1 = 0 \vee y_2 = 0 \text{ ist geometr. unmögl.})$

Sonderfall:
wenn P Wendepkt., dann $R = P$
(selbiges, wenn Q Wendepkt.)



$$x_1 \neq x_2$$

$$P + Q = -R$$

$$\begin{aligned} m &= (y_2 - y_1)/(x_2 - x_1) \\ x_3 &= m^2 - x_1 - x_2 \\ y_3 &= m(x_1 - x_3) - y_1 \end{aligned}$$

$$\begin{aligned} x_1 &= x_2 \\ 0 \neq y_1 \neq y_2 \neq 0 \end{aligned}$$

$$P + Q = 0$$

$$\begin{aligned} x_1 &= x_2 \\ y_1 &= y_2 = 0 \end{aligned}$$

$$P + Q = 0$$

$$\begin{aligned} x_1 &= x_2 \\ y_1 &= y_2 \neq 0 \end{aligned}$$

$$P + Q = -R$$

$$\begin{aligned} m &= (3x_1^2 + a)/(2y_1) \\ x_3 &= m^2 - x_1 - x_2 \\ y_3 &= m(x_1 - x_3) - y_1 \end{aligned}$$

Quadratischer Twist:

(VL: $F = \mathbb{F}_p$)

Sei $E_{a,b}$ eine EC über Körper F und $d \neq 0$ kein Quadrat in F .

Das Interessante ist, dass $E_{a,b}$ u. E_{d^2a, d^3b} zwar eine unterschiedliche Arithmetik haben, die komplexe Analyse der Kurven aber dieselbe ist!

Definiere den „Quadratischen Twist“: $E_{a,b}^{(d)} := E_{d^2a, d^3b}$. Während $E_{a,b}$ und $E_{a,b}^{(d)}$ nicht isomorph über F sind, sind sie es über einer Körpererweiterung von F , in der es ein u mit $u^2 = d$ gibt, bspw. im algebraischen Abschluss von F (VL: \mathbb{F}_p).

Wenn F ein endlicher Körper mit q Elementen (VL: $q \in \mathbb{P}$) ist, dann gilt $|E_{a,b}| + |E_{a,b}^{(d)}| = 2q + 2$, da man für jedes $x \in F$ genau 2 Punkte auf $E_{a,b}$ und/oder $E_{a,b}^{(d)}$ erhält (genaueres siehe Extrablatt).

Theorem von Hasse-Weyl / Hasse's theorem on elliptic curves:

...gibt eine obere und untere Abschätzung für die Anzahl der Punkte auf einer EC über einem endlichen Körper mit q Elementen (VL: $q \in \mathbb{P}$): $|E_{a,b}| - (q+1) \leq 2\sqrt{q}$ (d.h. EC $E_{a,b}$ hat $\approx q$ Elemente).

EC DH:

$G \in E_{a,b}$ ist öffentlich

$$A \xrightarrow{t_A \cdot G} B$$

$$A \xleftarrow{t_B \cdot G} B$$

Achtung! EC ist additive Gruppe, daher Mal statt Hoch!

Geheimnis: $(t_A \cdot t_B) \cdot G$

DLP: $t_A = d \log G (t_A \cdot G)$

ElGamal-Verschlüsselungsverfahren:

Alice's privater Schlüssel: $t_A \in N$

Alice's öffentl. Schlüssel: $A = t_A \cdot G, A \in E_{a,b}$

Bob's Nachricht an Alice: $N \in E_{a,b}$

Allgemein bekannt: $G \in E_{a,b}$, $\text{ord}(G) = q$

Verschlüsseln: Wähle zufällig $r \in \mathbb{Z}_{[2, q-2]}$

$$(t \cdot G, N + r \cdot A)$$

Entschlüsseln: $N = (N + r \cdot A) - t_A \cdot r \cdot G$

Klappt da $r \cdot A = t \cdot t_A \cdot G = t_A \cdot r \cdot G$

Schnorr-Signaturverfahren:

↳ Signieren, and. Darstellungen, Sicherheit
wg. SPH (ohnehim)

Sei E EC über Körper \mathbb{F}_p (p prim), $G \in E$, $\text{ord}(G) = n$ prim; öffentlich.

Alice: Privater Schlüssel: $d_A \in \mathbb{Z}/n\mathbb{Z}$

Öffentlicher Schlüssel: $P_A = d_A \cdot G \in E$

Nachricht: $m \in [1, n-1]$

Signatur berechnen:

[1] Wähle zufällig $k \in [1, n-1]$

$Q = k \cdot G = (x_Q, y_Q)$ temporär/Ephemeral-Schlüssel

[2] $r = \text{Hash}(m \parallel x_Q) \bmod n$

[3] $s = k - r \cdot d_A \bmod n$

(r, s) ist Alice's Signatur von m .

Bob: Signatur prüfen: $(x_Q', y_Q') = s \cdot G + r \cdot P_A = (s+r \cdot d_A) \cdot G = k \cdot G$ (*)

Prüfe, ob $r = \text{Hash}(m \parallel x_Q')$. (mod n)

⇒ Kein Invertieren nötig (gut für Chipkarten z.B.), dafür Hashfunktion.

Verschiedene Darstellungen von EC:

(any EC can be written in Weierstrass form)

- Weierstraß-Darstellung (bisher): $E_{a,b} = \{(x, y) \in \mathbb{F}^2 \mid y^2 = x^3 + ax + b\} \cup \{(0, 0)\}$
- Projektive Weierstrass-Koordinaten: $E_{a,b} = \{(X, Y, Z) \in \mathbb{F}^3 \mid (\frac{Y}{Z})^2 = (\frac{X}{Z})^3 + a(\frac{X}{Z}) + b\} \cup \{(0, 0, 0)\}$
- ↳ wobei $(k \cdot X, k \cdot Y, k \cdot Z) \stackrel{\sim}{=} (X, Y, Z)$
- ↳ in Sage bis auf $0 = (0, 0)$ auf $Z=1$ normiert mit $0 := (0, 1, 0)$; ansonsten normierte $Z=1$ (Sage)
- Montgomery-Darstellung/-Form-/Kurve: $E_{a,b} = \{(x, y) \in \mathbb{F}^2 \mid y^2 = (x+a)^3 + d(x+a) + b\} \cup \{(0, 0)\}$
- ↳ $a \in \mathbb{F}$ ist eine Nullstelle von x^3+ax+b
↳ geht n. für jede EC in Weierstraß-Form!
↳ $(\exists x : x^2 = 3x^2 + a$, d.h. ist quadrat. Rest)
- Edwards-Darstellung/-Kurve: $E_{a,d} = \{(x, y) \in \mathbb{F}^2 \mid ax^2 + y^2 = 1 + dx^2y^2\}$
- mit $d = \frac{A+2}{B}; d = \frac{A-2}{B}$ bzw. $A = \frac{2(d+d)}{d-d}$; $B = \frac{2}{d-d}$
- en.wiki: $x = \frac{u}{v}$; $y = \frac{u-1}{u+1}$ bzw. $u = \frac{1-y}{1+y}$; $v = \frac{1+y}{1-y}$
- und $(x_1, y_1) + (x_2, y_2) = \frac{(x_1y_2 + y_1x_2)}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - dx_1x_2}{1 - dx_1x_2y_1y_2}$

Wichtiger ist aber die...:

- Edwards-Darstellung/-Kurve:

Sicherheit von EC/DLP: → Weierstraß-Param. b fehlt in Formeln!

- SPH, Pollard- ρ , BSGS gibt es ohne hin (generisch); Indexkalkül dagegen n. für EC
- Invalid Curve Attack: Annahmen: - EC für Public Key Crypto eingesetzt, priv. Key = const.
(Biehl et al., 2000) (Jager et al.) - kann $k \cdot P$ für belieb. P berechnen - Server using Static ECPH

↳ TIS-ECPH: - Das Opfer (Server/Smartcard) überprüft n., ob $P \in E_{a,b}$.

en.wiki/ECPH: (TLS-)ECPH: Alice: $(d_A, Q_A = d_A \cdot G \text{ ephemeral})$; Bob: $(d_B, Q_B = d_B \cdot G \text{ static/trusted})$

Shared Secret: $d_A \cdot Q_B = d_B \cdot Q_A = (x_K, y_K)$ private/public $G = \text{agreed upon}$

Reguläres Szenario:

Alice → $Q_A = d_A \cdot G \rightarrow$ Server

Alice ← $Q_B = d_B \cdot G \rightarrow$ Server

Alice → $\text{Client Finished} \xrightarrow{A} \text{Server}$

Alice ← $\text{Server Finished} \xrightarrow{A} \text{Server}$
(mit $d_A \cdot Q_B = d_B \cdot Q_A$ als Schlüssel)

→ (spezielle Kurven meiden)

Eve → $P \notin E_{a,b} \rightarrow$ Server

Eve ← $Q_B = d_B \cdot G \rightarrow$ Server

Eve → $\text{Client Finished} \xrightarrow{A} \text{Server}$

(mit $d_B \cdot P$ als Schlüssel, geraten!)

Falls Eve ← $\text{Server Finished} \xrightarrow{A} \text{Server}$

dann kenne $d_B \bmod \text{ord}(P)$!

⇒ wiederhole & finde wann $d_B \bmod \text{ord}(P)$ mit CRT

(P = Generator einer kl. Untergruppe v. $\mathbb{Z}/p\mathbb{Z}$ -ordnung)

(Meier, Okamoto, Vanstone)

- MFLV-Angriff: Sei Abb. w: $E_{a,b}(\mathbb{F}_p) \times E_{a,b}(\mathbb{F}_p) \rightarrow \mathbb{F}_{p^k}$ bilinear, k Einbettungsgrad.

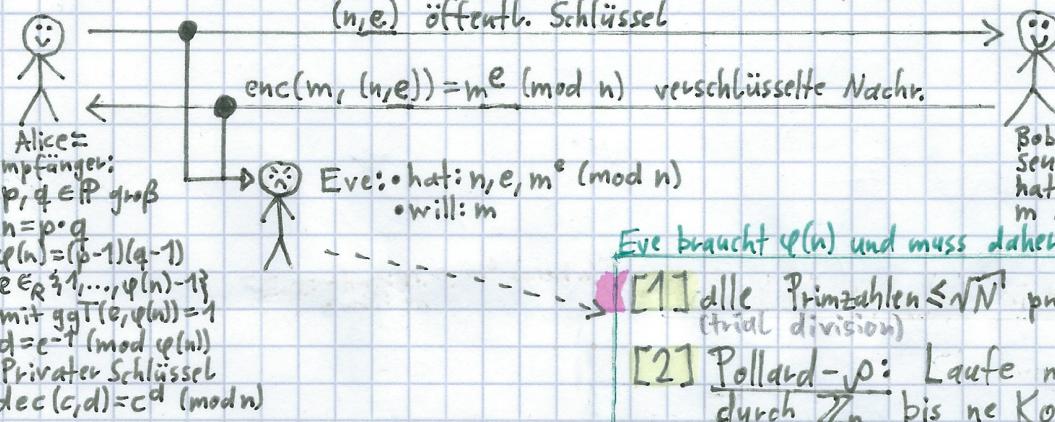
Gegeben: $x \cdot P$; Gesucht: x Verschiebe nun das DLP-Problem nach \mathbb{F}_{p^k} :

Es gilt: $w(x \cdot P, Q) = w(P, Q)^x \Rightarrow x = \text{dLog}_w(P, Q)$ (w heißt "Weil-Paarung")

⇒ Lehre: verwende nur Kurven mit hohem Einbettungsgrad, am besten standardisierte EC

Eigenes „Gesamtbild“ CRYPTO1 – „Faktorisierung I + II“ 1. / 2. / 3. / 14. 6. 22 & 16. 6. & 21. 6.

RSA (Rivest, Shamir, Adleman) zw. Alice & Bob:



Eve braucht $\varphi(n)$ und muss daher n faktorisieren:

[1] alle Primzahlen $\leq \sqrt{N}$ probieren $\Rightarrow O(\sqrt{N})$ (trial division)

[2] Pollard - p : Laufe mit x_i, y_i zufällig durch \mathbb{Z}_n bis ne Kollision mod p , d.h. $x_i \equiv y_i \pmod{p} \Rightarrow p \mid x_i - y_i \Rightarrow \text{ggT}(x_i - y_i, N) = p \Rightarrow O(\sqrt{N})$

[3] Pollard - $p-1$: Annahme: $p-1$ ist B -potenzglatt, d.h. die maximale Primzahlpotenz, die $p-1$ teilt, ist $\leq B$.

$$M(B) := \prod_{\substack{i=1 \\ p_i \in \mathbb{P}}} \mathbb{Z}_{p_i^{e_i}} \text{ mit } p_i, e_i \in \mathbb{B} \text{ und } e_i \text{ maximal}$$

$$\text{Dann gilt } p-1 \mid M(B).$$

Außerdem gilt für ein $a \in \mathbb{Z}_p^*$: $a^{p-1} \equiv 1 \pmod{p}$

$$\Rightarrow p = \text{ggT}(a^{p-1} - 1, N)$$

$$= \text{ggT}(a^{M(B)} - 1, N)$$

[4] ECM/Lenstra (3. schnellste bekannte Meth.): Wählt E ab über \mathbb{Z}_n u. $P_0 \in E$ ab zufällig. $P_i = 2 \cdot P_{i-1}$; Punktverdopplung involviert Berechnung von $m = \frac{3x_i^2 + a}{2y_i}$; irration „khalft“ weil Nenner nicht invertierbar ist; mit allergrößter Wk. teilt dann p oder q diese Zahl $\Rightarrow O(\exp((\sqrt{2} + o(1)) \sqrt{\ln n \ln \ln n}))$

[5] Fermat ($p \neq q$): Beob.: $p \cdot q = \frac{1}{4}[(p+q)^2 - (p-q)^2]$ kombi $\Leftrightarrow \left(\frac{p+q}{2}\right)^2 - N = \left(\frac{p-q}{2}\right)^2 \Leftrightarrow t^2 - N = s^2$; hier mit $t_i = \lceil \sqrt{N} \rceil$; $t_i = t_{i-1} + 1$; sobald $t_i^2 - N$ trial division! Quadratzahl, faktorisiere: $N = t_i^2 - s^2 = (t_i - s)(t_i + s)$

[6] Quadratisches Sieb (#2.): Durchsiebt $t_1^2 - N, \dots, t_k^2 - N$ aus Fermat nach B-glatten Zahlen: $t_i^2 - N = p_1^{e_1} \cdot \dots \cdot p_s^{e_s}$ Löse ein LGS in y_1, \dots, y_k über \mathbb{Z}_2 , sodass: (erhalten) $(t_1^2 - N) y_1, \dots, (t_k^2 - N) y_k = T^2$. $S := t_1 y_1 \cdot \dots \cdot t_k y_k$. $S^2 = T^2 \pmod{N} \Rightarrow N \mid (S-T) \cdot (S+T)$. Falls $N \nmid (S-T) \wedge N \nmid (S+T)$: $\text{ggT}(N, S+T) \in \{p, q\}$ $\Rightarrow O(\exp((1+o(1)) \sqrt{\ln n \ln \ln n}))$

[7] Zahlkörper sieb (1. schnellste Meth.) \Rightarrow VL nur Skizze.

By the way geht das Ganze auch umgedreht: $\varphi(n) = (p-1)(q-1) = pq - p - q + 1 = n - p - q + 1 \Rightarrow p + q = n - \varphi(n) + 1$

Damit lassen sich die Nullstellen von $x^2 - [n - \varphi(n) + 1]x + n$ (Es geht auch $= x^2 - (p+q)x + pq$ $\Rightarrow p, q \in \mathbb{Z}$) bestimmen; bei diesen handelt es sich um p, q .

$\mathbb{Z}_n^* =$
invertierbare
Elemente
in \mathbb{Z}_n

Wie findet Alice große Primzahlen?:

Strategie: • Wähle $x \in \mathbb{Z}/N+1$ zufällig & groß.
• Wende Primzahltest an (s.u.).
↳ Nein: $x \leftarrow x+2$.

Abstrakter Primzahltest (für n):

Idee: $\mathbb{Z}_n^* = \mathbb{Z}_n \setminus \{0\}$ gdw. n prim
↳ Beweis: $n \in \mathbb{P} \Rightarrow \mathbb{Z}_n^* = \{x \mid \text{ggT}(x, n) = 1\} = \{1, \dots, n-1\}$

$n \notin \mathbb{P} \Rightarrow n = a \cdot b = 0 \Rightarrow$ Nullteiler

Test: Konstruiere $L_n \subseteq \mathbb{Z}_n \setminus \{0\}$ mit Eig.:

[1] $x \in L_n$ schnell prüfbar

[2] $n \in \mathbb{P} \Rightarrow L_n = \mathbb{Z}_n^* = \mathbb{Z}_n \setminus \{0\}$

[3] $n \notin \mathbb{P} \Rightarrow \exists c \in \mathbb{Z}: |L_n| \leq c \cdot (n-1)$

Dann: 1.) Wähle $x \in \mathbb{Z}_n \setminus \{0\}$ zufällig

$\mathbb{Z}_n \setminus \{0\}$ • 2.) Prüfe mit [1], ob $x \in L_n$.

L_n • 3.) Ja: Wiederhole 1.)

Nein: $n \notin \mathbb{P}$ wegen [2]

Wegen [3] gilt für jedes $x: P[x \in L_n] = \frac{|L_n|}{n-1} \leq c$ und damit insg. $P[x_1, \dots, x_t \in L_n] \leq c^t$, d.h. mit Anzahl Schritte t groß genug ist Wk. für falsch-positive Antw. beliebig klein

Konkreter Primzahltest #1: Fermat

$L_n := \{x \in \mathbb{Z}_n \setminus \{0\} \mid x^{n-1} = 1\} \subseteq \mathbb{Z}_n^*$ (klar)

QS

[1] ✓ wegen Square & Multiply

[2] ✓ da Element hoch Gruppenord. immer 1

[3] ✓ wenn n eine Carmichael-Zahl, dann $|L_n| = \varphi(n) \leq n$ (kleinst: 561) (für Carmichael-Zahlen)

Konkreter Primzahltest #2: Miller-Rabin

$L_n := \{x \in \mathbb{Z}_n \setminus \{0\} \mid x^{n-1} = 1 \wedge \forall 0 \leq i \leq h-1: x^{2^i} \equiv 1 \pmod{n} \Rightarrow x^{2^h} \equiv 1 \pmod{n}\}$

Vor.: n ist ungerade: $n-1 = 2^h \cdot m$, m ungerade

[1] $[x^m, (x^m)^2, (x^m)^4, (x^m)^8, \dots, (x^m)^{2^h}] \equiv 1$

enthält nur Einsen; d.h. $n-1$ als Element??!

[2] $n \in \mathbb{P}: x^2 \equiv 1 \pmod{n} \Rightarrow n \mid x^2 - 1 \Rightarrow x \equiv \pm 1$

[3] $n \notin \mathbb{P}: |L_n| \leq \frac{1}{4} (n-1)$, d.h. $\frac{1}{4} \cdot (n-1)$ (aufr. 2.2.1.)

Warum gilt $\text{dec}(c, d) = c^d \pmod{n} = m$?

$\Leftrightarrow m^d \equiv m \pmod{n}$

$\Leftrightarrow m^d = (m^{\varphi(n)})^k \cdot m \pmod{n}$

$\Leftrightarrow ed = k \cdot \varphi(n) + 1 \Leftrightarrow d = e^{-1} \pmod{\varphi(n)}$

n ist Carm.
-Zahl gdw.
 $d^{-1} \equiv 1$
(mod n)

Für alle a mit

$\text{ggT}(n, a) = 1$

für alle a mit

$\text{ggT}(n, a) \neq 1$

gilt $a^{-1} \neq 1$

da sonst a

invertierbar wäre!

Element

hoch

Gruppen-

ordnung = 1

Eigenes „Gesamtbild“ CRYPTO1 – Quantencomp. (I)

- Normaler Computer: Bit:
- Quantencomputer: Qubit:

Statys Qubit:
= Vektor der Länge 1
in \mathbb{C}^2

2 Basiszustände: 0 und 1 (sonst nix)
2 Basiszustände $|0\rangle := \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ und $|1\rangle := \begin{pmatrix} 0 \\ 1 \end{pmatrix} \in \mathbb{C}^2$
Sowie Überlagerung dieser $\Rightarrow VR \subset \mathbb{C}^2$:
 $\alpha \cdot |0\rangle + \beta \cdot |1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}; |\alpha|^2 + |\beta|^2 = 1; \alpha, \beta \in \mathbb{C}$
 \Rightarrow kann man n. sehen; Messen: $|0\rangle$ mit Wk. $|\alpha|^2$
 $|1\rangle$ mit Wk. $|\beta|^2$

de.wiki: $\delta \neq$ Qubit ist zstd. $|0\rangle$ mit Wk. $|\alpha|^2$ u. in $|1\rangle$ Wk. $|\beta|^2$.
 \rightarrow das könnte auch ein klassischer Computer

- Nun mehrere Qubits: auch durch VR beschrieben: $\mathbb{C}_2 \otimes \mathbb{C}_2 \otimes \mathbb{C}_2 \otimes \dots$

Seien V, W 2 VR über demselben Körper F .

Das Tensorprodukt $V \otimes W$ ist ein VR mit Basis $\{v \otimes w \mid v \in V, w \in W\}$.

Das Tensorprodukt $x \otimes y$ zweier Vektoren $x \in V, y \in W$ ist definiert als:

$$x \otimes y := \sum_{b \in B_V} \sum_{c \in B_W} x_b y_c b \otimes c$$

Damit ist das Tensorprodukt eine bilineare Abb. $V \times W \rightarrow V \otimes W$.

2 Qubits: $\mathbb{C}^2 \otimes \mathbb{C}^2$ mit Basis $|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle$
bzw. $|00\rangle, |01\rangle, |10\rangle, |11\rangle$

bzw. $|0\rangle, |1\rangle, |2\rangle, |3\rangle$ \hookrightarrow kennt Dim. n. mehrfach

3 Qubits: $\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2$ mit Basis $|0\rangle \otimes |0\rangle \otimes |0\rangle, \dots, |1\rangle \otimes |1\rangle \otimes |1\rangle$
 \Rightarrow Der VR für n Qubits hat 2^n Dimensionen u. Basis $|0\rangle, \dots, |2^n-1\rangle$

Man kann leicht nachrechnen, dass sich die Normierung auf 1 auf $\mathbb{C}^2 \otimes \dots \otimes \mathbb{C}^2$ überträgt!

- Nun operiere auf einem solchen System von n Qubits:

Dies geschieht mit einer $2^n \times 2^n$ Matrix U .

bei Multiplikation,

Diese ist unitär, d.h. $U^* U = U U^* = I$. Sie erhält damit sowohl Norm als auch Skalarprodukt.

Auf 1 Qubit: NOT:

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \xrightarrow{|0\rangle \mapsto |1\rangle, |1\rangle \mapsto |0\rangle} \boxed{x} \quad (\text{"Pauli-X-Gatter"})$$

(Notation VL)

\hookrightarrow $t = \text{konjugiert}$
 $T = \text{transponiert}$

Siehe auch Wikis:

Liste der Quantengatter: $|0\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ HADAMARD: $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

$|1\rangle \mapsto \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$ \hookrightarrow konst., in = gew. gemischten Zustand.

T -Gatter: $\begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$ \hookrightarrow Phasengatter

Auf 2 Qubits: CNOT: $q_0 \xrightarrow{\text{---}} q_1 \xrightarrow{\boxed{x}} q_0 \xrightarrow{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}$ d.h. $|00\rangle \mapsto |00\rangle, |10\rangle \mapsto |10\rangle, |01\rangle \mapsto |11\rangle, |11\rangle \mapsto |10\rangle$

Kontrolliertes T-Gatter: $q_0 \xrightarrow{\text{---}} q_1 \xrightarrow{\boxed{T}} q_0 \xrightarrow{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\pi/4} \end{pmatrix}}$ $|11\rangle \mapsto e^{i\pi/4} \cdot |11\rangle, |00\rangle \mapsto |00\rangle$, etc...

- Die Quanten-Fourier-Transformation: = ein wesentlicher Bestandteil des Shor-Algorithmus

= eine Zerlegung der diskreten Fourier-Trafo in unitäre Matrizen, sodass als Q.schaltkreis implementierbar

= Quanten-Analog der diskreten Fourier-Trafo $n = \# \text{Qubits}$

\rightarrow kann effizient auf Q.comp. durchgeführt werden mit $O(n^2)$ (Hadamard-) Gattern

\hookrightarrow die klassische diskrete Fourier-Trafo braucht $O(n \cdot 2^n)$ Gatter, exp. mehr

n Qubits, $N = 2^n$ $w_N = e^{(2\pi i)/N}$ \hookrightarrow als Zahlen multipliziert!

QFT als Abb. auf Basisvektoren: $|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} w_N^{xy} |y\rangle$

QFT als Matrix: $\frac{1}{\sqrt{N}} \begin{pmatrix} w_N^{0,0} & w_N^{0,1} & \dots & w_N^{0,N-1} \\ w_N^{1,0} & w_N^{1,1} & \dots & w_N^{1,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_N^{N-1,0} & w_N^{N-1,1} & \dots & w_N^{N-1,N-1} \end{pmatrix} = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & w_N & \dots & w_N^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & w^{N-1} & \dots & w^{(N-1)} \end{pmatrix}$

Inverse QFT als Abb.: $|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} w_N^{-xy} |y\rangle$

- Idee / Intro Shor-Algorithmus: Sei $a \in \mathbb{Z}_{p \cdot q}$ mit $\text{ggT}(a, p \cdot q) = 1$.

Iteriere die Abb. $f_a: \mathbb{Z}_{pq} \rightarrow \mathbb{Z}_{pq}, x \mapsto ax$: $f_a^k: a \mapsto a^k \cdot x$

$k = \text{ord}(a) \Rightarrow f_a^k = \text{id}$ \rightarrow ist eine Schwingung mit Basisfrequenz = $\text{ord}(a)$

\rightarrow analysiere mit Fourier-Trafo die Frequ.anteile heraus!

Habe $a^k \equiv 1 \pmod{N}$ gefunden $\Rightarrow N \mid (a^k - 1) \Rightarrow N \mid (a^{k/2} - 1)(a^{k/2} + 1) \Rightarrow p = \text{ggT}(N, a^{k/2} - 1)$

Ziel: analysiere Harmonie der Zahl a "

BRUNNEN

Eigenes „Gesamtbild“ CRYPTO1 - Quantencomp. (II) / Shor (bisher: Bauteile)

(VL vom 29.6.22)

23.8.22 & 24.8.22

Faktorisierter N:

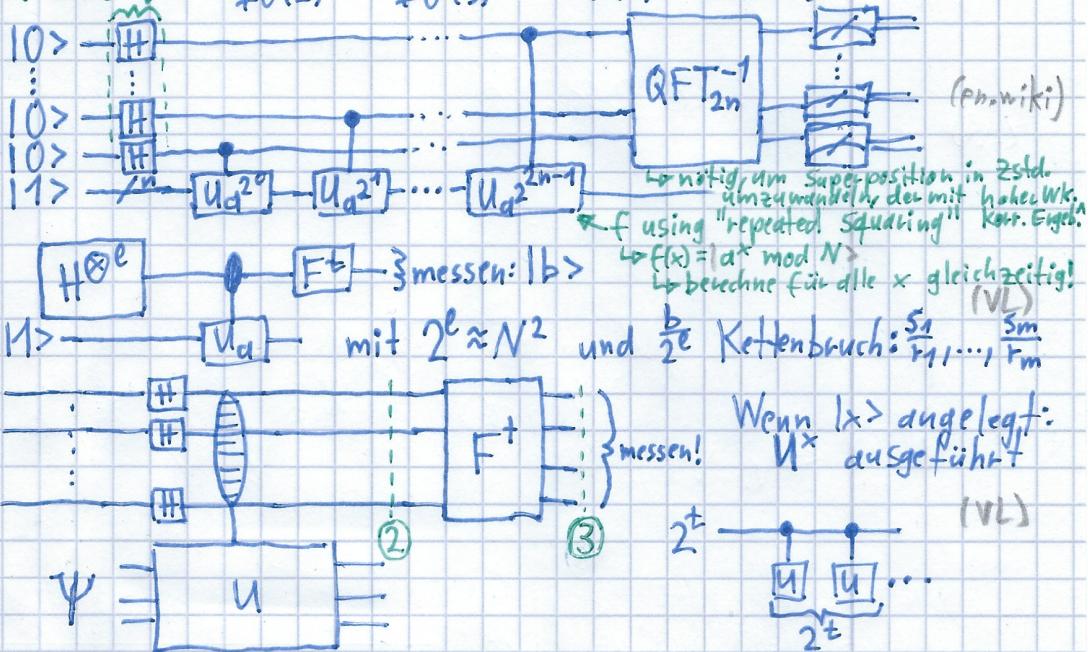
- (1) Wähle eine Zahl $1 < d < N$ mit $\text{ggT}(d, N) = 1$ (ausgenommen schon fertig).
- (2) Bestimme mithilfe des Quantenteils das kleinste $r \in \mathbb{N}$ mit $d^r \equiv 1 \pmod{N}$.
- (3) Falls r ungerade oder $d^{r/2} \equiv -1 \pmod{N}$ ($\text{charf} \equiv 1$): wiederhole (1).
- (4) Gebe $\text{ggT}(d^{r/2}-1, N)$ als Lösung zurück.
(oder $\text{ggT}(d^{r/2}+1, N)$)
muss (!) nicht triviale Teiler von N enthalten, da:

$$(d^{r/2}-1) \cdot (d^{r/2}+1) = d^r - 1$$

„create a superposition of states“

Quantenteil:

“custom designed
for each choice
of N and each
choice of the
random a ”



e Qubits

Wenn $|x\rangle$ angelegt:
 U_x ausgeführt

Wie funktioniert der Quantenteil?:

/

Eigenes „Gesamtbild“ CRYPTO1 - Gitterbasierte Verf. (\mathbb{F})

Problem, das wir ein neues Problem brauchen.“ (RSTdD durch Q.comp. gekn.)

[1] Was ist ein Gitter / „Lattice“?!

Ein n -dimensionales Gitter ist eine Teilmenge $L \subseteq \mathbb{R}^n$ mit n linear unabhängigen Vektoren (b_1, \dots, b_n) , sodass alle $x \in L$ als ganzzahlige Linearkombination von b_i darstellbar sind: $x = \sum_{i=1}^n z_i b_i$; $z_i \in \mathbb{Z}$.
z.B. $b_1 = (1, 0), b_2 = (0, 1)$; $L = \text{span}(b_1, b_2) = \{(x_1, x_2) \in \mathbb{Z}^2 \mid x_1, x_2 \in \mathbb{Z}\}$.

(Idee: verhüftige Basis = priv. Schl.)

(Idee: unrechtfertige Basis = öff. Schl.)

[2] 1. Gitterproblem: CVP (Closest Vector Problem)

Gegeben: $x \in \mathbb{R}^n$; Gesucht: Der (oder die) nächstgelegene Gitterpunkt?!

[3] 2. Gitterproblem: SVP (Shortest Vector Problem)

Finde einen Vektor $\neq 0$ in L mit der kürzesten Länge (d.h. alle and. mind. so lang)

[4] Bsp.: $b_1 = (5, 7), b_2 = (8, 11)$, nächstgel. Gitterpkt. zu $(1, 0, 0, 9)$ ist $(1, 1)$!

[5] Babai: Löse CVP zu $x \in \mathbb{R}^n$: $x = \sum_{i=1}^n x_i b_i$, $x_i \in \mathbb{R}$ (x als reeller Vektor in Vektorbasis)

$x = \sum_{i=1}^n \lceil x_i \rceil b_i \in L$ (simpler: runde auf nächste ganze Zahl)

→ Bsp.: Gute Basis: $(1, 0), (0, 1)$: $\lceil 1.0 \rceil \cdot (1, 0) + \lceil 0.9 \rceil \cdot (0, 1) = (1, 1)$ \square
Schlechte Basis: $(5, 7), (8, 11)$: $\lceil -3.8 \rceil \cdot (5, 7) + \lceil 2.5 \rceil \cdot (8, 11) = (-4, -6) \neq (1, 1) \quad \nabla$
 $\lceil -3.8 \rceil + 2.5 \lceil 8 \rceil = \lceil 1.0 \rceil \quad \nabla$
 $\Rightarrow 2$ hin zur Null runden bei 5)

[6] Für eine orthogonale Basis funktioniert dieses Heuris. von Babai: Bew.:
 $\|x-y\|^2 = \|\sum_{i=1}^n x_i b_i - \sum_{i=1}^n y_i b_i\|^2 = \langle x-y, x-y \rangle = \langle \sum_{i=1}^n (x_i - y_i) b_i, \sum_{j=1}^n (x_j - y_j) b_j \rangle$
 $= \sum_{i,j=1}^n (x_i - y_i)(x_j - y_j) \in \mathbb{Z} \quad \nabla$
 $\Rightarrow \sum_{i,j=1}^n (x_i - y_i)^2 \|b_i\|^2$ ist minimal gdw. $y_i = \lceil x_i \rceil$ linear
orthogonal basis

[7] Babai für „fast“ orthogonale Basen löst das Approximate CVP/CVP-NF.

[8] Babai für schlechte Basen funktioniert nicht. (siehe obiges Bsp.)

[9] Ein Gitter L mit Basis (b_1, \dots, b_n) hat die Generatormatrix (b_i) (d.h. einfach zeilenweise Basisvektoren)

Nenne Determinante dieser Matrix Det. des Gitters \Rightarrow betragsmäßig eindeutig $\neq 0$ (egal)

Außerdem gilt: $\det(L) = |\det(G(L))| = \text{vol}(F)$ mit Fundamentalbereich F

Nennen „unimodular“. $(F(b_1, \dots, b_n)) := \{x = \sum x_i b_i \mid 0 \leq x_i \leq 1\}$; Sind G_1 und G_2 zwei Generatormatrizen von L , sind ganzzahlig. dann existiert eine Matrix U mit $G_1 = U \cdot G_2$ u. $\det(U) = \pm 1 \Rightarrow$ Äquiv. Klassen \Rightarrow gleiche Vertreter:

„Unimodular“: [10] Hermittische Normalform: (ist ein solcher Vertreter UND effektiv zu berechnen)

Sei G eine Generatormatrix, U eine unimodulare Matrix, $U \cdot G$ in oberer Dreiecksform mit Werten > 0 in der Diagonale und ≥ 0 darüber; $U \cdot G$ heißt Hermitt. Normalform.

[11] Prättisch heißt das, man darf: • Zeilen vertauschen (eingeschränkter Gauß-Algo)

Sage: matrix([22, [[1, 1], [1, 0]]]).echelon_form() • Vielfache einer Zeile auf andere addieren

• mit ± 1 durchmultiplizieren

Bsp.: $\begin{pmatrix} 5 & 7 \\ 8 & 11 \end{pmatrix} \sim \begin{pmatrix} 5 & 7 \\ 3 & 4 \end{pmatrix} \sim \begin{pmatrix} 2 & 3 \\ 3 & 4 \end{pmatrix} \sim \begin{pmatrix} 2 & 3 \\ 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \sim \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \sim \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \text{Hermitt. Norm. form!} \quad \nabla$

(Das Produkt unimodulärer Matrizen ist wieder eine unimodulare Matrix!) \Rightarrow d.h. kriegt Generatormatrix leicht in Norm. NF \Rightarrow Probl. für alle, die was verbauen wollen

(„Bruch“) [12] Hadamard-Ratio: Sei G Generatormatrix mit Zeilen b_1, \dots, b_n .

$0 \leq H(G) = H(b_1, \dots, b_n) = \frac{\det(G)}{\sqrt{\|b_1\| \cdot \dots \cdot \|b_n\|}} \leq 1$ Hadamardsche Ungleichung \rightarrow Algo.

Richtig Es gilt: $H(G) = 1 \Leftrightarrow (b_1, \dots, b_n)$ orthogonal & je größer $H(G)$, desto besser klappt Brötchen

CVP $\hat{=} [13]$ GGtt-Kryptosystem (Goldreich-Goldwasser-Halevi): = „gescheitert“

öffentl. Parameter: $n \in \mathbb{N}$ Dimension, $\delta > 0$ klein („Fehler-Range“), N („Nachr.-Range“)

privater Schlüssel: Generatormatrix G von L mit $H(G)$ nahe an 1 („geheim/gute Basis“)

öffentl. Schlüssel: Generatormatrix A von L mit $H(A)$ nahe an 0 („schlechte Basis“)

Verschlüsseln: $m \in [-v, v]^n \cap \mathbb{Z}^n$ Nachr., $e \in (-\delta, \delta)^n$ Fehlerv., $ENC(m) = m \cdot A + e \notin L$

Entschlüsseln: $c = ENC(m)$; best. mit dabei nächstgeleg. Gitterpkt.: $m \cdot A + e \in L$ \square

Eigenes „Gesamtbild“ CRYPT01 – Gitterbasierte Verf. (II) 27.8.22 & 31.11.

Öffentl. schlechte Basis \rightarrow Gemeinsame HNF $\xrightarrow{?}$ Geheime gute Basis \rightarrow lösbar CVP m. B. bei
 \Rightarrow "Gibt tatsächlich so ein Verfahren, aber schwierig den so durchschlagend."
Sei (b_1, \dots, b_n) die Basis eines Gitters (mit ganzzahligen Koord.) $L \subseteq \mathbb{Z}^n$.
 \Leftrightarrow muss nicht sein!

Orthonormalisierung =
wenn zusätzlich
normiert
(würde)

[1]

Gram-Schmidt Orthogonalisierung (Verfahren):

... baut effektiv aus Basis eine Orthogonalsbasis, nur leider wahrsch. n. mehr im Gitter:

$$b_1^* = b_1 \quad (b_1, \dots, b_n) \quad (b_1^*, \dots, b_n^*)$$

$$b_2^* = p_2 - N_{2,1} b_1^* \text{ mit } N_{2,1} = \frac{\langle b_2, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} \Rightarrow \langle b_1^*, b_2^* \rangle = 0 \text{ (orthogonal)}$$

$$b_3^* = p_3 - N_{3,1} b_1^* - N_{3,2} b_2^* \text{ mit } N_{3,1} = \frac{\langle b_3, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle}, N_{3,2} = \frac{\langle b_3, b_2^* \rangle}{\langle b_2^*, b_2^* \rangle}, \dots$$

$$b_k^* = p_k - N_{k,1} b_1^* - N_{k,2} b_2^* - \dots - N_{k,k-1} b_{k-1}^* \text{ mit } N_{k,j} = \frac{\langle b_k, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}, j < i, N_{i,i} = 1$$

$$b_n^* = \dots$$

\Leftrightarrow Aber was nun, wenn ich $|N_{k,j}|$ runde, um im Gitter zu verbleiben?:

$$\Leftrightarrow \text{Bsp.: } b_1 = \begin{pmatrix} 5 \\ 7 \end{pmatrix}; b_2 = \begin{pmatrix} 8 \\ 11 \end{pmatrix}; \langle b_2, b_1^* \rangle = 5 \cdot 8 + 7 \cdot 11 = 117 \neq 0, \text{ d.h. } p. \text{ orthogonal}$$

$$b_1^* = \begin{pmatrix} 5 \\ 7 \end{pmatrix}; b_2^* = \begin{pmatrix} 8 \\ 11 \end{pmatrix} - \frac{11}{7} \begin{pmatrix} 5 \\ 7 \end{pmatrix} = \begin{pmatrix} 8 - 585/74 \\ 11 - 819/74 \end{pmatrix} = \begin{pmatrix} 7/74 \\ -5/74 \end{pmatrix}; \langle b_1^*, b_2^* \rangle = 5 \cdot \frac{7}{74} + 7 \cdot \frac{-5}{74} = 0 \quad \checkmark$$

Man sieht allerdings leicht, dass i. A. nun $z_1 \cdot b_1^* + z_2 \cdot b_2^* \notin L$:

[2] Selbes Verfahren, nur runde $|N_{k,j}|$, um im Gitter zu verbleiben. ("Reduktion")

$$\Leftrightarrow \text{Bsp.: } \lceil -117/74 \rceil = -2; b_1 = \begin{pmatrix} 5 \\ 7 \end{pmatrix}; b_2 = \begin{pmatrix} 8 \\ 11 \end{pmatrix} - 2 \begin{pmatrix} 5 \\ 7 \end{pmatrix} = \begin{pmatrix} -2 \\ 3 \end{pmatrix};$$

$b_1 = \begin{pmatrix} 5 \\ 7 \end{pmatrix}$ u. $b_2 = \begin{pmatrix} -2 \\ 3 \end{pmatrix}$ nun zwar Basis v. L , aber $\langle b_1, b_2 \rangle = -31 \neq 0$!

[3] / [4] Nicht nur Reduzieren, sondern auch Tauschen:

\Leftrightarrow denn (Zitat Skript): "Sometimes the reduction algorithm produces shorter vectors when we switch two basis vectors b_i and b_{i+1} ."

\Leftrightarrow LLL-Algorithmus (A. Lenstra, H. Lenstra, L. Lovász):

\Leftrightarrow erster effizienter Gitterreduktionsalgorithmus (bezogen auf eukl. Norm)

\Leftrightarrow berechnet für ein Gitter eine Basis aus möglichst kurzen Vektoren

```
def LLL_reduction(matrix = (b1, ..., bn), δ ∈ [1/4, 1] = 3/4):
    n = len(matrix.rows())
    for i in range(1, n):
        (b1, ..., bn), {b_i: *, 3 N_{i,j}:} = reduction(matrix = (b1, ..., bn))
        for j in range(0, n-1):
            if not Lovasz-condition(b_i: *, b_{i+1}: *, N_{i+1,i}: δ) // L-Bed. v. gilt
                b_i, b_{i+1} = b_{i+1}, b_i // Vertausche
                continue outer loop
            break
    return (b1, ..., bn), {b_i: *}, {N_{i,j}}
```

"irre" $\not\leq$ (1.) Dieser Algorithmus terminiert? \Leftrightarrow aber nur für $\delta \in (1/4, 1)$!
(2.) Er hat sogar polynomiale Laufzeit: $n^{6 \cdot (\ln(\max \|b_i\|))} \not\leq$

```
def Lovasz-condition(b_i: *, b_{i+1}: *, N_{i+1,i}: δ): // für alle 1 ≤ i < n
    return δ · \|b_i\|^2 ≤ \|b_{i+1}\|^2 + 2 N_{i+1,i} · \|b_i\|^2
    ⇔ return (δ - N_{i+1,i}^2) \|b_i\|^2 ≤ \|b_{i+1}\|^2 // mit {b_i: *} den zugehörigen GSO-Daten
```

„unendliche Liste“: (3.) Eine δ -LLL-reduzierte Basis (b_1, \dots, b_n) hat folgende Eigenschaften:
(a) Der erste Vektor b_1 kann nicht viel größer sein als der kürz. Vektor $\neq 0$:
für $\delta = \frac{3}{4}$ im Speziellen: $\|b_1\| \leq 2^{(n-1)/2} \cdot \lambda_1(L)$
(b) Außerdem ist b_1 durch die Determinante des Gitters beschränkt:
für $\delta = \frac{3}{4}$ im Speziellen: $\|b_1\| \leq 2^{(n-1)/4} \cdot (\det(L))^{1/n}$
(c) Das Produkt der Normen der Basisv. kann n. viel größer sein als die Det. des Gitters:
für $\delta = \frac{3}{4}$: $\prod_{i=1}^n \|b_i\| \leq 2^{(n-1)/4} \cdot \det(L)$
(d) $\min \|b_i\| \leq \lambda_1(L)$ (e) $\|b_i\| \leq 2^{(i-1)/2} \|b_i:*\|$, $1 \leq j \leq n$
(f) $\|b_i:*\|^2 \leq \|b_i\|^2 \leq \frac{1}{2} + 2^{i-2} \cdot \|b_i:*\|^2$, $1 \leq i \leq n$ (g) $\|b_i:*\|^2 \leq 2 \cdot \|b_{i+1}:*\|^2$, $1 \leq i \leq n$
(VL): $\det(L) \leq \lambda_1^n(L)$ Notation: $\lambda_1(L) := \max_S \left(\min_{b \in S} \|b\| \right)$, $S \subseteq L$, $|S| = i$; $\lambda_1^n(L)$ kürz. Vektor $\neq 0$ in L
(VL): (m.M.n. andersrum!) $\lambda_1(L) := \max_S \left(\min_{b \in S} \|b\| \right)$, $S \subseteq L$, $|S| = i$; $\lambda_1^n(L)$ kürz. Vektor $\neq 0$ in L

Eigenes „Gesamtbild“ CRYPTO1 – Post-Quanten-Algo

27.8.22

(Schneller als NTRU)

(GGH
war
CVP*)

„Purdue geflogen“

NTRU (Encrypt) \cong SVP!

(beide Gitter-basiert:)

\approx GGH/CVP \rightarrow NIST

(z.B.
 $N=1087$)

= 1, n-th truncated polynomial ring
• Polynomring $R = \mathbb{Z}[x]/\langle x^N - 1 \rangle; N \in \mathbb{P}$ (groß)
↳ d.h. $x^N - 1 = 0 \Rightarrow x^N = 1, x^{N+1} = x$, usw.

(d.h. Grad
überschreitet nie N)

(z.B. $p=3$)

• Bruder #1: $R_p = \mathbb{Z}_p[x]/\langle x^N - 1 \rangle; p \in \mathbb{P}$ (klein)

(z.B. $p=7681 \in \mathbb{P}$)

↳ d.h. Koeff. noch mod p , ansonsten dasselbe

(z.B. $q=2^10$)

• Bruder #2: $R_q = \mathbb{Z}_q[x]/\langle x^N - 1 \rangle; N \in \mathbb{Q}, p \in \mathbb{Q}$

(WIKI: $q=2048$)

• d, sodass $p(6d+1) < q$ (*)

de.wiki
erst im
Abschnitt
„Effizienz“

• $T(a,b) := \begin{cases} a \text{ Koeff. von } f = 1 \\ b \text{ Koeff. von } f = -1 \\ N-a-b \text{ Koeff. v. } f = 0 \end{cases}$

Key Gen:

π = Projektion

• $f \in T(d+1, d)$ ternäres Polynom (geheim)

• $g \in T(d, d)$ ternäres Polynom (Ephemeral)

• Dabei muss $\pi_p(f)$ in R_p und $\pi_q(f)$

in R_q invertierbar sein, sonst generiere f neu!

↳ $f_p := \pi_p(f)^{-1} \in R_p$, d.h. $f \cdot f_p \equiv 1 \pmod{p}$

↳ $f_q := \pi_q(f)^{-1} \in R_q$, d.h. $f \cdot f_q \equiv 1 \pmod{q}$

• Öffentl. Schlüssel: $h = f_q \cdot \pi_q(g) \in R_q$

Für nur damit schneller

Encrypt:

• Nachricht $m \in \lambda_p(R_p) \subseteq R$ Polynom

↳ $\lambda_p: \mathbb{Z}_p \rightarrow \mathbb{Z}, z_1, z_2, \dots, z_d = 2p, 3p, \dots, p, 4p, 5p, \dots, 2p, \dots$

$\in [0, p)$ $\mapsto \xi \in [-\frac{p}{2}, \frac{p}{2}]$ „symmetr. Vertreter“

↳ $\lambda_p: R_p \rightarrow R$ Koeffizientenweise

• Ternäres Polynom $t \in T(d, d) \subseteq R$

• Ciphertext: $c = p \cdot \pi_q(t) \cdot h + \pi_q(m) \pmod{q}$

• $a = \pi_q(f) \cdot c \in R_q \pmod{q}$

• $m' = \pi_p(\lambda_q(a)) \cdot f_p \in R_p \pmod{p}$

• $m = \lambda_p(m') \in \lambda_p(R_p) \subseteq R$

Decrypt:

• $q = \pi_q(f) \cdot c$

$= \pi_q(f) \cdot [p \cdot \pi_q(r) \cdot h + \pi_q(m)]$

$= p \cdot \pi_q(r) \cdot \pi_q(f) \cdot f_q \cdot \pi_q(g) + \pi_q(f) \cdot \pi_q(h) \cdot \pi_q(m)$

$= p \cdot \pi_q(r) \cdot [f \cdot f_q \cdot \pi_q(g) + \pi_q(f) \cdot \pi_q(h) \cdot \pi_q(m)]$

$= \pi_q(p \cdot r \cdot g + f \cdot m)$

↳ $\text{odd } \pi_q$ eine strukturerhaltende Abb.

↳ $\lambda_q(a) = p \cdot r \cdot g + f \cdot m$ (wegen *)

• $m' = \pi_p(\lambda_q(a)) \cdot f_p$

$= \pi_p(p \cdot r \cdot g + f \cdot m) \cdot f_p$

$= \pi_p(f \cdot m) \cdot f_p \quad (\text{da mod } p)$

$= \pi_p(f) \cdot \pi_p(m) \cdot f_p \quad (\text{da } f_p := \pi_p(f)^{-1})$

$\Rightarrow \lambda_p(m') = \lambda_p(\pi_p(m)) = m, m \in \lambda_p(R_p)$

Wo Gitter?:

• Ring $R = \text{Polynome mit Grad} \leq N-1$

$\Leftrightarrow \mathbb{Z}^N$ (Gitter)

• NTRU-Angriff: aus $h = f_q \cdot g$

• SVP-Angriff: f und g berechnen

(f, g) $\in \mathbb{Z}^N$ geheim: f und g

$(f, g) \in \mathbb{Z}^N$ \Leftrightarrow f geheim, g offen

$(f, g) \in \mathbb{Z}^N$ \Leftrightarrow f geheim, g offen

\Leftrightarrow f geheim, g offen