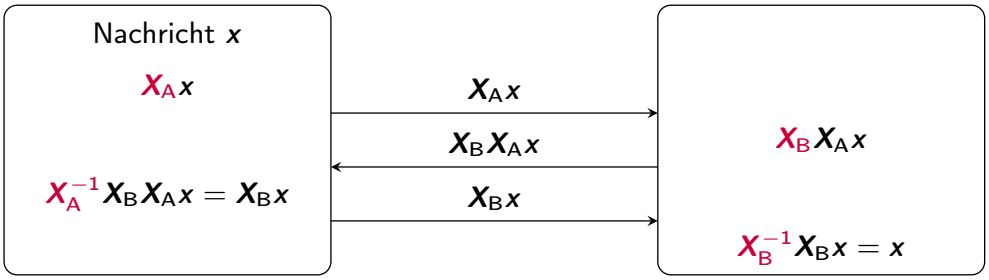
7.3 Unizitätslänge

- ein Chiffrierverfahren sei über Z verschiedene Schlüssel parametrierbar
- also könnte ein Geheimtext zu (höchstens) Z verschiedenen Klartexten entschlüsselt werden – wie viele davon sind sinnvoll?
- bzw. falls Länge des Klartextes k eindeutig: wie groß muss k werden, damit Entschlüsselung eindeutig? $\Rightarrow$ Unizitätslänge U
- Anzahl sinnvoller Texte der Länge k ungefähr $2{,}3^k$
- Wahrscheinlichkeit, dass zufälliger Text der Länge k sinnvoll: $\frac{2{,}3^k}{26^k} = \left(\frac{2{,}3}{26}\right)^k$
- Anzahl falscher, aber sinnvoller Texte der Länge k, die durch Entschlüsselung eines Geheimtextes mit Z verschiedenen Schlüsseln zu erwarten sind: $\left(\frac{2{,}3}{26}\right)^k \cdot (Z-1) \approx \left(\frac{2{,}3}{26}\right)^k \cdot Z$
- Entschlüsselung vermutlich eindeutig, wenn $\left(\frac{2{,}3}{26}\right)^k \cdot Z < 1$, also $k > \frac{\operatorname{ld} Z}{\operatorname{ld} 26 - \operatorname{ld} 2{,}3} \approx 0{,}29 \operatorname{ld} Z$
- Für VIGENÈRE der Periode d: $\operatorname{ld} Z = \operatorname{ld} 26^d = d \operatorname{ld} 26$, also $U \approx 1{,}34 \cdot d \Rightarrow$ ist der Schlüssel kürzer als $\frac{1}{1{,}34} \approx 74\,\%$ des Textes eindeutig entschlüsselbar!?
  - Betrachtung zu sehr vereinfacht: Textteil mit nicht-wiederholtem Schlüssel vermutlich uneindeutig, Textteil mit wiederholtem Schlüssel überbestimmt
  - aber spätestens ab $k = 2 \cdot d$ eindeutig

8 Schlüsselaustausch und öffentliche Schlüssel

- Chiffriersicherheit erfordert lange, häufig wechselnde Schlüssel
- Schlüsselverteilung ist gravierendes praktisches Problem
- es geht auch ohne vorherigen sicheren Schlüsselaustausch!
- naiver Ansatz: seien $\boldsymbol{X}_\mathrm{A}, \boldsymbol{X}_\mathrm{B}$ kommutative Verschlüsselungsverfahren, d.h. $\boldsymbol{X}_\mathrm{A}\boldsymbol{X}_\mathrm{B} = \boldsymbol{X}_\mathrm{B}\boldsymbol{X}_\mathrm{A}$



Problem: Klartext-Geheimtext-Kompromittierung von $\boldsymbol{X}_\mathrm{A}$ und $\boldsymbol{X}_\mathrm{B}$
Beispiel VIGENÈRE $\boldsymbol{X}_\mathrm{A}x = x + \boldsymbol{t}_\mathrm{A}, \boldsymbol{X}_\mathrm{B}x = x + \boldsymbol{t}_\mathrm{B}$:

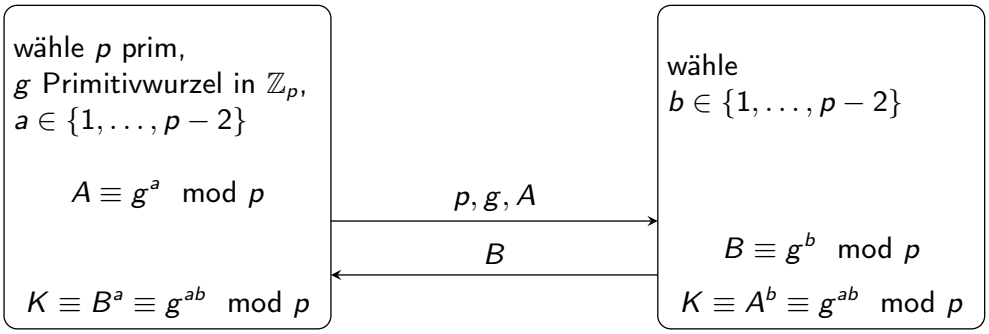
$$\boldsymbol{X}_\mathrm{B}^{-1}\boldsymbol{X}_\mathrm{A}^{-1}\boldsymbol{X}_\mathrm{B}\boldsymbol{X}_\mathrm{A}x = x$$

aber auch

$$\boldsymbol{X}_\mathrm{A}x - \boldsymbol{X}_\mathrm{B}\boldsymbol{X}_\mathrm{A}x + \boldsymbol{X}_\mathrm{B}x = x$$

8.1 Diffie-Hellman-Schlüsselaustausch

- Diffie-Hellman verwendet kein kommutatives Verschlüsselungsverfahren, sondern sichere Schlüsselvereinbarung



- Schlüssel K kann anschließend für geeignetes Verschlüsselungsverfahren verwendet werden
- Sicherheit beruht auf Schwierigkeit, diskreten Logarithmus zu berechnen, also a (bzw. b) aus p, g und A (bzw. B) zu bestimmen
- Primitivwurzel: $g^k \mod p, k \in \mathbb{N}$ erzeugt alle Elemente aus $\{1, \ldots, p-1\}$
  Beispiel: $p = 13, g = 7$

k	1	2	3	4	5	6	7	8	9	10	11	12
$g^k \mod p$	7	10	5	9	11	12	6	3	8	4	2	1

8.2 Effizientes Potenzieren mit dem Horner-Schema

- Sicherheit gegen Brute-Force-Angriffe erfordert Wahl von g und a mit Hunderten bis Tausenden bit Länge
- Potenz g^a wird (vor der Modulo-Operation) gigantisch groß
- wird die Potenz schrittweise ausgeführt, Modulo-Operation nach jeden Schritt möglich, aber $a - 1$ Multiplikationen nötig
- effizienter: Horner-Schema
- Beispiel:

$$\begin{aligned} 7^{11} &= (7^5)^2 \cdot 7 = ((7^2)^2 \cdot 7)^2 \cdot 7 = (((1 \cdot 7)^2 \cdot 1)^2 \cdot 7)^2 \cdot 7 \\ &\equiv (10^2 \cdot 7)^2 \cdot 7 \equiv (9 \cdot 7)^2 \cdot 7 \equiv 11^2 \cdot 7 \equiv 4 \cdot 7 \equiv 2 \mod 13 \end{aligned}$$

- Entschiedung, ob Multiplikation mit Basis hängt vom Bitmuster des Exponenten ab: $11 = 1011_2$
- Anzahl der Multiplikationen proportional zu $n = \lceil \mathrm{ld}\, a \rceil$, Länge des Exponenten in bit

8.3 Asymmetrische Verfahren und öffentliche Schlüssel

- Diffie-Hellman erlaubt sichere Schlüsselvereinbarung, authentifiziert aber nicht den Kommunikationspartner $\Rightarrow$ Risiko eines "Man-in-the-Middle"-Angriffs
- Lösung: asymmetrische Verfahren, $\boldsymbol{X}_\mathrm{A}^{-1}$ nicht (ohne unpraktikabel hohen Rechenaufwand) aus $\boldsymbol{X}_\mathrm{A}$ bestimmbar
- Alice veröffentlicht $\boldsymbol{X}_\mathrm{A}$
- Bob verschlüsselt Nachricht für Alice mit $\boldsymbol{X}_\mathrm{A}$
- nur Alice kennt $\boldsymbol{X}_\mathrm{A}^{-1}$ und kann die Nachricht entschlüsseln
- Zugehörigkeit von $\boldsymbol{X}_\mathrm{A}$ zu Alice muss nur einmal über fälschungssicheren Kanal verifiziert werden; keine Abhörsicherheit für Schlüsselaustausch erforderlich
- Rückweg genauso mit Bobs Schlüsselpaar $\boldsymbol{X}_\mathrm{B}$ und $\boldsymbol{X}_\mathrm{B}^{-1}$
- Kommunikation von K Teilnehmern erfordert K Schlüsselpaare (im Gegensatz zu $K(K-1)/2$ symmetrischen Schlüsseln)
- auch Signieren möglich, falls $\boldsymbol{X}_\mathrm{A}^{-1}$ und $\boldsymbol{X}_\mathrm{A}$ kommutativ: Alice verschlüsselt mit $\boldsymbol{X}_\mathrm{A}^{-1}$; jeder kann mit $\boldsymbol{X}_\mathrm{A}$ entschlüsseln, aber erfolgreiche Entschlüsselung beweist, dass Verschlüsselung von Alice

- Diffie-Hellman-Verfahren lässt sich erweitern zu asymmetrischem Verfahren ELGAMAL
- wie vorher werden p prim, g als Primitivwurzel und a zufällig gewählt (von jedem Teilnehmer)
- öffentlicher Schlüssel: (p, g, A) mit $A \equiv g^a \mod p$
- privater Schlüssel: (p, a)
- zu verschlüsselnde Nachricht $x \in \{1, \ldots, p-1\}$
- verschlüsselte Nachricht: (B, c) mit $B \equiv g^b \mod p$, $c \equiv Kx \mod p$, $K \equiv A^b \mod p$ und zufällig gewählten b (also Diffie-Hellman-Schlüsselaustausch und Verschlüsselung durch Multiplikation mit K)
- Entschlüsselung: bestimme $K \equiv B^a \mod p$, damit Rekonstruktion des Klartextes $K^{-1}c \mod p$
- Hinweis: kleiner Satz von Fermat erlaubt direkte Berechnung von $K^{-1} \equiv B^{p-1-a} \mod p$

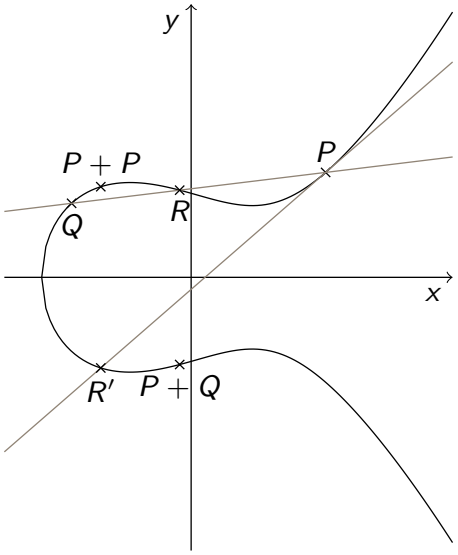
- bereits vor ELGAMAL schlugen Rivest, Shamir und Adleman ein asymmetrisches Verfahren vor
- für das RSA-Verfahren wählt jeder Teilnehmer zwei (große) Primzahlen p', p'' und $e \in \{1, \ldots, \Phi(N)\}$, wobei $N = p'p''$ und $\Phi(p'p'') = (p'-1)(p''-1)$, sodass e in $\mathbb{Z}_{\Phi(N)}$ invertierbar ist
- öffentlicher Schlüssel: (N, e)
- privater Schlüssel: (N, d), wobei d Inverses zu e in $\mathbb{Z}_{\Phi(N)}$
- zu verschlüsselnde Nachricht $x \in \{1, \ldots, N-1\}$
- Verschlüsselung: $y \equiv x^e \mod N$
- Entschlüsselung: $\hat{x} \equiv y^d \mod N$
- es folgt $\hat{x} = (x^e)^d = x^{ed} = x^{1+k\Phi(N)} = x \cdot x^{k\Phi(N)}$ mit $k \in \mathbb{N}$
- es lässt sich zeigen, dass $x^{\Phi(N)} \equiv 1 \mod N$, also $\hat{x} = x$
- Berechnung von d erfordert $\Phi(N)$ erfordert p' und p'', aber Primfaktorzerlegung (zu) rechenaufwändig, daher praktisch unmöglich, wenn p' und p'' unbekannt

8.4 Asymmetrische Verfahren in der Praxis: OpenPGP

- OpenPGP ist Nachrichtenstandard für mit asymmetrischen Verfahren chiffrierte Nachrichten, insbesondere für Emails
- u.a. von der Open-Source-Software GNU Privace Guard (GnuPG) implementiert; Integration in Thunderbird mit Plugin Enigmail
- zu verschlüsselnde Nachricht wird zunächst komprimiert (ZIP oder ZLIB)
- es wird ein symmetrisches Verfahren (Triple-DES, IDEA, CAST5, BLOWFISH, ...) ausgewählt und für dieses ein zufälliger Schlüssel gebildet
- die (komprimierte) Nachricht wird mit dem symmetrischen Verfahren verschlüsselt
- für jeden gewünschten Empfänger wird der symmetrische Schlüssel um zufällige Bits erweitert und asymmetrisch mit dem öffentlichen Schlüssel des Empfängers verschlüsselt (RSA oder ELGAMAL)
- die symmetrisch verschlüsselte Nachricht und die asymmetrisch verschlüsselten Schlüssel bilden endgültige Geheimnachricht
- für Signaturen wird ein Hash (MD5 oder SHA-1) der Nachricht gebildet und nur dieser signiert (mit DSA oder RSA)

8.5 Elliptische Kurven

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



definiere Addition für Punkte $P = (x_P, y_P)$ und $Q = (x_Q, y_Q)$ der Kurve

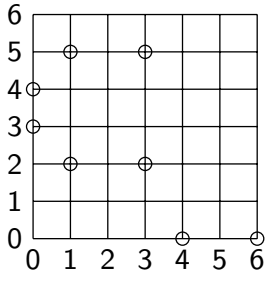
- falls $x_P \neq x_Q$: bestimme dritten Schnittpunkt R der Geraden P-Q mit Kurve und spiegele an x-Achse:

$$\begin{aligned} x_R &= s^2 - x_P - x_Q \\ y_R &= y_P + s \cdot (x_R - x_P) \\ \text{mit } s &= (y_Q - y_P)(x_Q - x_P)^{-1} \\ P + Q &= (x_R, -y_R) \end{aligned}$$

- falls $P = Q, y_P \neq 0$: bestimme Schnittpunkt R' Tangente an P mit der Kurve und spiegele an x-Achse::

$$\begin{aligned} x_{R'} &= s^2 - 2x_P \\ y_{R'} &= y_P + s \cdot (x_R - x_P) \\ \text{mit } s &= (3x_P^2 + a)(2y_P)^{-1} \\ P + P &= (x_{R'}, -y_{R'}) \end{aligned}$$

- für Summe von Punkten mit $x_P = x_Q$ wird 0 als "Punkt im unendlichen" ergänzt; damit
  - für $P \neq Q, x_P = x_Q$: $P + Q = 0$
  - für $P = Q, y_P = 0$: $P + P = 0$
- es ergibt sich eine Gruppe mit 0 als neutralem Element:
  - $P + 0 = 0 + P = P$ für alle P
  - Addition ist assoziativ, $(P + Q) + R = P + (Q + R)$ für alle P, Q, R
  - zu jedem P gibt es ein Inverses $Q = -P$ (durch Spiegelung an x-Achse), sodass $P + Q = Q + P = 0$
- Multiplikation mit $n \in \mathbb{N}$ als n-fache Addition: $nP = P + \cdots + P$
- effizient auszuführen durch Verdopplungen und Additionen entsprechend Horner-Schema
- mit denselben Rechenregeln übertragbar, wenn Koordinaten der Punkte in $\mathbb{Z}_p$, p prim



$y^2 \equiv x^3 + x + 2 \mod 7$

- Multiplikation $Q = nP$ relativ einfach berechenbar
- Umkehrung (P und Q gegeben, finde n) wesentlich rechenaufwändiger
- Verfahren wie Diffie-Hellman oder ELGAMAL auf elliptische Kurven übertragbar
  - Alice wählt primes Modul p, Parameter a, b der elliptischen Kurve, einen Punkt P und eine Zahl n
  - Alice berechnet $Q = nP$, veröffentlicht alles außer n
  - Bob wählt eine Zahl m, berechnet $R = mP$ und sendet es an Alice
  - Alice berechnet Schlüssel $K = nR = mnP$, Bob berechnet Schlüssel $K = mQ = mnP$
- Vorteil gegenüber Potenzierung im Restklassenring: Unterschied Rechenaufwand zwischen Verschlüsselung und Inversion der Potenzierung bzw. Multiplikation bei elliptische Kurven größer $\Rightarrow$ gleiche Sicherheit bei kürzeren Schlüsseln, weniger Rechenaufwand
- Einsatz deshalb insbesondere bei SmartCards, z.B. neuer Personalausweis, aber auch für SSL/TLS vorgesehen

8.6 Web of Trust und PKI

- größte praktische Schwierigkeit bei asymmetrischer Kryptographie: Sicherstellen, dass der öffentliche Schlüssel zum gewünschten Empfänger gehört (kein Man-in-the-Middle)
- Beglaubigung von Schlüsseln durch Notare – Signieren mit asymmetrischer Verschlüsselung
  - Alice erzeugt Schlüsselpaar und legt ihren öffentlichen Schlüssel bei Trent vor
  - Trent prüft Alice' Identität und verifiziert, dass er tatsächlich ihren Schlüssel erhalten hat, dann signiert er eine Nachricht bestehend aus ihrem Schlüssel und ihrem Namen (signierter Schlüssel oder Zertifikat)
  - Bob hat Trents öffentlichen Schlüssel über einen vertrauenswürdigen Kanal erhalten und er vertraut Trent; bekommt er Alice' öffentlichen Schlüssel mit Trents Signatur, vertraut er daher darauf, dass der Schlüssel tatsächlich zu Alice gehört
- Ansatz von PGP/GnuPG: Web of Trust
  - Schlüssel können mehrere Signaturen tragen
  - Nutzer wählt selbst aus, wem er vertraut; zwei Stufen: volles oder teilweises Vertrauen
  - Nutzer definiert, wann er einem Schlüssel vertraut; z.B. zwei Signaturen von Nutzern denen er voll vertraut, oder fünf Signaturen von Nutzern, denen er teilweise vertraut
- alternative: Public Key Infrastructure (PKI)
  - Schlüssel wird nur einmal signiert (Zertifikat)
  - Zertikate werden von Zertifizierungsstellen (CA) ausgestellt
  - Schlüssel der CAs werden mit Betriebssystem oder Software (z.B. Browser) ausgeliefert