Kryptographie und Komplexität

Einheit 4.3

Angriffe auf das RSA Verfahren

1. Faktorisierungsangriffe
2. Andere Angriffe
3. Richtlinien für die Schlüsselauswahl

SICHERHEIT DES RSA VERFAHRENS

- **Sicherheit des geheimen Schlüssels**
  - Einziger bekannter Weg, Schlüssel zu brechen, ist Faktorisierung von n
  - Man kann zeigen, daß beide Probleme äquivalent sind (Beweis folgt)
  - Wie aufwendig ist Faktorisierung?
    - $\mathcal{NP}$-Problem: Standardsuche nach Teilern ist exponentiell in $\|n\|$
    - Primzahltests liefern nur Information, aber keine Faktoren
    - Es gibt Algorithmen, die den Exponenten stark verkleinern
    - Trotzdem braucht man für 1024 Bit mehr als 100.000.000 Jahre
- **Semantische Sicherheit**

  Kann ein Schlüsseltext dechiffriert werden, ohne des Schlüssel zu kennen?
  - Bis heute nicht abschließend geklärt - es gibt Probleme in Einzelfällen
  - Äquivalenz von RSA zum Faktorisierungsproblem nicht nachweisbar

  Wie leicht können RSA-chiffrierte Nachrichten manipuliert werden?
  - Man kann Schlüsseltexte kombinieren ohne den Klartext zu kennen

Sicherheit des RSA Schlüssels $K=(n,p,q,d,e)$

Berechnung des geheimen Schlüssels d aus e, n ist genauso schwer wie die Faktorisierung von $n = p{\cdot}q$

- **Faktorisierung bricht den geheimen Schlüssel**
  - Wenn Eve n in p und q zerlegen kann, dann kann sie auch $d = e^{-1} \bmod (p-1)(q-1)$ in $\mathcal{O}(\|n\|^2)$ Schritten berechnen
- **Der geheime Schlüssel liefert eine Faktorisierung**

  Sei $s = \max\{t \in \mathbb{N} \mid 2^t \text{ teilt } ed-1\}$, $k = (ed-1)/2^s$ und $a \in \{1...n-1\}$ beliebig

  1. Ist $gcd(a,n)=1$ so ist $\text{order}_{\mathbb{Z}_n^*}(a^k)=2^i$ für ein $i \leq s$

     Es ist $(a^k)^{2^s} = a^{ed-1} \equiv 1 \bmod n$, also ist $\text{order}_{\mathbb{Z}_n^*}(a^k)$ Teiler von 2^s

  2. Ist $gcd(a,n)=1$, $\text{order}_{\mathbb{Z}_p^*}(a^k) \neq \text{order}_{\mathbb{Z}_q^*}(a^k)$ so ist $gcd(a^{2^t k}-1, n) \in \{p,q\}$ für ein $t<s$

     Wie oben sind $\text{order}_{\mathbb{Z}_p^*}(a^k)$ und $\text{order}_{\mathbb{Z}_q^*}(a^k)$ Teiler von 2^s.

     Sei o.B.d.A. $\text{order}_{\mathbb{Z}_q^*}(a^k) = 2^t < \text{order}_{\mathbb{Z}_p^*}(a^k) \leq 2^s$. Dann gilt

     $(a^k)^{2^t} \equiv 1 \bmod q$ aber $(a^k)^{2^t} \not\equiv 1 \bmod p$ also $gcd(a^{2^t k}-1, n) = q \neq 1$

Sicherheit des RSA Schlüssels $K=(n,p,q,d,e)$

'RP' Algorithmus zur Faktorisierung von n mithilfe von K

– Wähle $a \in \{1...n-1\}$ zufällig

– Ist $g = gcd(a,n) \neq 1$ dann ist g echter Teiler von n

– Ansonsten setze $s = \max\{t \in \mathbb{N} \mid 2^t \text{ teilt } ed-1\}$ und $k = (ed-1)/2^s$

– Wenn $gcd(a^{2^t k}-1, n) = q \neq 1$ für ein $t = s-1, s-2, ..., 0$, dann ist q Teiler von n

---

Für zusammengesetzte n ist die Anzahl der $a<n$ mit $gcd(a,n)=1$ und $\text{order}_{\mathbb{Z}_p^*}(a^k) \neq \text{order}_{\mathbb{Z}_q^*}(a^k)$ ist mindestens $\varphi(n)/2$

– Nach dem chinesischen Restsatz gibt es $g<n$, das $\mathbb{Z}_p^*$ und $\mathbb{Z}_q^*$ erzeugt.

– Falls $e = \text{order}_{\mathbb{Z}_p^*}(g^k) > \text{order}_{\mathbb{Z}_q^*}(g^k)$ dann sei $0<x<p$ ungerade, $y \leq q-2$ und a Lösung der Kongruenzen $a \equiv g^x \bmod p$, $a \equiv g^y \bmod q$.
Da e Zweierpotenz ist, ist $\text{order}_{\mathbb{Z}_p^*}(a^k) = e/gcd(e,x) = e > \text{order}_{\mathbb{Z}_q^*}(a^k)$
Da g Erzeugende von $\mathbb{Z}_p^*$ und $\mathbb{Z}_q^*$ ist, gibt es für jedes x, y ein anderes a.

– Falls $\text{order}_{\mathbb{Z}_p^*}(g^k) = \text{order}_{\mathbb{Z}_q^*}(g^k)$ dann sei $0<x<p$ ungerade, $y \leq q-2$ gerade (bzw. umgekehrt) und $a<n$ mit $a \equiv g^x \bmod p$, $a \equiv g^y \bmod q$.
Es folgt $\text{order}_{\mathbb{Z}_p^*}(a^k) \neq \text{order}_{\mathbb{Z}_q^*}(a^k)$ für $2 \cdot (p-1)(q-1)/4$ Zahlen.

Wahrscheinlichkeit, in r Iterationen keinen Teiler zu finden, ist 2^{-r}

Es gibt viele Arten von Faktorisierungsangriffen

- **Probedivision**
  – Standardverfahren, gut für kleine Faktoren
- **Methoden für spezielle Zahlen**
  – Pollard $p{-}1$: Für Faktor p hat $p{-}1$ nur kleine Primfaktoren
  – Fermat-Methode: Faktoren liegen nahe bei $\sqrt{n}$
- **Pollard ρ**
  – Systematische Suche nach Kollisionen $x \equiv x' \bmod p$ für unbekanntes p
- **Methoden auf Basis quadratischer Kongruenzen**
  – Dixon Random Squares, Quadratische Siebe, Zahlkörpersiebe
- **Elliptische-Kurven-Faktorisierung**
  – Probabilistischer Algorithmus auf Basis elliptscher Kurven

$\vdots$ $\vdots$

Probedivision

- **Einfacher, leicht zu programmierender Ansatz**
  - Alle Teiler der Zahl n werden der Reihe nach durchgetestet
  - Hochgradig ineffizient — Laufzeit $\mathcal{O}(n) = \mathcal{O}(2^{\|n\|})$
- **Optimierungen haben geringen Effekt**
  - Außer 2 nur noch ungerade Zahlen betrachten — $\mathcal{O}(2^{\|n\|})$
  - Suche beschränkt auf Zahlen bis $\lfloor\sqrt{n}\rfloor$ — $\mathcal{O}(2^{\|n\|/2})$
  - Beschränke Suche auf Primzahlen mit dem Sieb des Erathostenes
    Komplexität ist $\mathcal{O}(\sqrt{n}/ln(\sqrt{n}))$ $= \mathcal{O}(2^{\|n\|/2-\log\|n\|})$
- **Nur geeignet für Zahlen mit kleinen Teilern**
  - Suche nach Teilern muß auf Schranke B begrenzt werden
  - Schranke jenseits von 10^7 wenig sinnvoll

Zahlen größer als 10^{14} sollten anders faktorisiert werden

Pollard $p{-}1$ Algorithmus

Nur für Zahlen mit bestimmten Eigenschaften

- **Wenn $p{-}1$ für ein $p|n$ nur kleine Primfaktoren hat**
  – Sei k ein beliebiges Vielfaches von $p{-}1$
  – Nach Satz von Fermat ist $a^k \bmod p = 1$ für jedes a mit $p \not| a$
  – Da p Teiler von $a^k{-}1$ ist, muß $gcd(a^k{-}1, n)$ echter Teiler von n sein, wenn $a^k{-}1$ kein Vielfaches von n ist

  Wie bestimmt man das Vielfache einer unbekannten Zahl?
  – Wenn die Primfaktorzerlegung von $p{-}1$ nur aus Primzahlpotenzen $q^e{\leq}B$ für eine Schranke B bestehen, dann ist $B!$ Vielfaches von $p{-}1$
- **Einfacher Algorithmus**
  – Wähle $a := 2$
  – Berechne $a' := a^{\prod_{j=2}^{B} j} \bmod n$
  – Falls $d := gcd(a'{-}1, n){>}1$, dann ist d Faktor von n
- **Komplexität abhängig von B** $\quad\mathcal{O}(B{\cdot}\|B\|{\cdot}\|n\|^2)$
  – B modulare Potenzierungen mit $j{\leq}B$ und Berechnung des gcd

Pollard $p-1$: Ablaufbeispiel

- **Faktorisierung von** $n = 6609029$
  - Primfaktoren bis $B = 3$ reichen aus, um Faktoren $p = 7$ und $q = 944147$ zu finden
- **Faktorisierung von** $n = 891404116139$
  - Primfaktoren bis $B = 47$ reichen aus, um Faktoren $p = 944137$ und $q = 944147$ zu finden
  - Da beides Primzahlen sind, benötig Probedivision nahezu 10^6 Schritte
- **Faktorisierung von** $n = 263185729$
  - Primfaktoren bis $B = 8111$ müssen untersucht werden, um Faktoren $p = q = 16223$ zu finden
  - Probedivision ist nahezu genauso schnell

Pollard ρ Algorithmus

- **Suche $x \neq x' \in \mathbb{Z}_n$ mit $1 < gcd(x-x', n) < n$**
  - Ist p Primfaktor von n so gilt $p \leq gcd(x-x', n) < n$ falls $x \equiv x' \bmod p$
  - Bei einer Teilmenge $X \subseteq \mathbb{Z}_n$ mit $1.2\sqrt{p}$ Elementen findet man eine solche Kollision mit Wahrscheinlichkeit 50% (Geburtstagsparadox)
  - Überprüfen aller Paare aus X braucht mehr als $p/2$ gcd-Berechnungen
- **Erzeuge und prüfe Zufallselemente schrittweise**
  - Berechne Folge $x_1, x_2, \ldots$ mit $x_{k+1} := f(x_k) \bmod n$ (f Zufallspolynom)
  - Gilt $x_i \equiv x_j \bmod p$ für ein $i<j$, dann gilt auch $f(x_i) \equiv f(x_j) \bmod p$ also $x_{i+1} \equiv x_{j+1} \bmod p$ und damit $x_{i+k} \equiv x_{j+k} \bmod p$ für alle k

    Folge der x_k läuft in eine Schleife, was aussieht wie ein $\boldsymbol{\rho}$
  - Hat die Schleife die Länge $l = j-i$, so gibt es ein $k \in \{i..j-1\}$, das Vielfaches von l ist. Für dieses k gilt $x_k \equiv x_{2k} \bmod p$
- **Einfaches Suchverfahren**
  - In Schritt k bestimme $x := f^k(x_1)$, $x' := f^{2k}(x_1)$ und $d := gcd(x'-x, n)$
  - Ist $d>1$, dann ist d Faktor von n
  - Ist $d=n$ oder $k = B$, so breche ohne Erfolg ab

POLLARD ρ: ABLAUFBEISPIEL

- **Trace der Faktorisierung von $n = 275831$**

```
Schleife 1.          x = 1         x′ =2  d = 1
Schleife 2.          x = 2        x′ =26  d = 1
Schleife 3.          x = 5  x′ =182499  d = 1
Schleife 4.         x = 26    x′ =6145  d = 1
Schleife 5.        x = 677   x′ =26256  d = 1
Schleife 6.   x = 182499  x′ =187948  d = 1
Schleife 7.   x = 119245  x′ =104247  d = 1
Schleife 8.     x = 6145  x′ =260046  d = 1
Schleife 9.   x = 248010  x′ =252849  d = 1
Schleife 10.   x = 26256  x′ =153840  d = 1
Schleife 11.   x = 75868   x′ =89454  d = 1
Schleife 12. x = 187948   x′ =10831  d = 1
Schleife 13. x = 153690  x′ =244353  d = 1
Schleife 14. x = 104247  x′ =141598  d = 1
Schleife 15. x = 247272  x′ =230974  d = 1
Schleife 16. x = 260046  x′ =191915  d = 1
Schleife 17.   x = 90833   x′ =89356  d = 1
Schleife 18. x = 252849  x′ =266080  d = 101
```

  – Faktoren sind 101 und 2731 (beides Primzahlen)

- **Faktorisierung von $n = 891404116139$**

  – 1410 Schritte nötig um Faktoren $p = 944137$, $q = 944147$ zu finden

Fermat-Faktorisierungsmethode

Gut wenn Differenz der Faktoren gering

- **Suche Faktoren p, q nahe bei $\sqrt{n}$**
  - Sei $n = p{\cdot}q$ mit $p<q$ ungerade (nicht notwendigerweise prim)
  - Da $q-p$ gerade ist, sind $x = (p+q)/2$ und $d = (q-p)/2$ ganze Zahlen und es gilt $n = p{\cdot}q = (x-d)(x+d) = x^2-d^2$
  - Damit ist x^2-n Quadratzahl mit der Eigenschaft $x > \lfloor\sqrt{n}\rfloor$
- **Einfacher Suchalgorithmus**
  - Suche das erste $x>\lfloor\sqrt{n}\rfloor$ für das $x^2-n =: d^2$ Quadratzahl ist
  - $p := x-d$ und $q := x+d$ sind die Faktoren von n
- **Ablaufbeispiel**
  - Für $n = 891404116139$ ist $x_0 := \lfloor\sqrt{n}\rfloor = 944141$
  - Es gilt $(x_0+1)^2 = 891404116164$, also $(x_0+1)^2-n = 25$
  - Die beiden Faktoren sind $p = 944137$ und $q = 944147$

Faktorisierung mit quadratischen Kongruenzen

Erweiterte Idee der Fermat-Faktorisierung

- **Suche nichttriviale $x, y \in \mathbb{Z}_n$ mit $x^2 \equiv y^2 \bmod n$**
  - Ist n Teiler von $x^2-y^2 = (x-y)(x+y)$ und $x \not\equiv \pm y \bmod n$
    dann haben n und $x-y$ sowie n und $x+y$ gemeinsame Teiler
  - Suche liefert Faktoren von n
- **Beispiel: Kongruenzen für $n = 15770708441$**
  - Suche liefert $x = 125979$ und $y = 10000$
    Dann ist $x^2 = 15870708441 = n + y^2$ und $gcd(x-y, n) = 115979$
    Damit ist 115979 Faktor von n (ebenso wie $gcd(x+y, n) = 135979$)
- **Suche x und y ausgehend von $\lfloor\sqrt{n}\rfloor$**
  - Anders als bei Fermat muß $x^2 \bmod n$ Quadratzahl modulo n sein
    und der Abstand von x zu $\lfloor\sqrt{n}\rfloor$ kann sehr groß werden

Wie kann man x und y systematisch finden?

Bestimmung quadratischer Kongruenzen

- **Erzeuge x und y durch Kombination von Quadratzahlen**
  - Zerlege gewisse $b_i = x_i^2 \bmod n$ in Primfaktoren
  - Suche Kombination der Zerlegungen, die eine Quadratzahl ergeben: bestimme Werte $z_1, .., z_k$, so daß alle Primfaktoren p_j von $b_1^{z_1} \cdot b_2^{z_2} .. b_k^{z_k}$ geradzahlig vorkommen, d.h. $b_1^{z_1} \cdot b_2^{z_2} .. b_k^{z_k} = \Pi_{j=1}^m p_j^{2e_j}$
  - Setze $x = x_1^{z_1} \cdot x_2^{z_2} .. x_k^{z_k} \bmod n$ und $y = \Pi_{j=1}^m p_j^{e_j}$
- **Methode: Lösung linearer Gleichungssysteme**
  - Ist $b_i = \prod_{j=1}^m p_j^{e_{i,j}}$ dann ist $b_1^{z_1} \cdot b_2^{z_2} .. b_k^{z_k}$ genau dann eine Quadratzahl, wenn die Summe der entstehenden Exponenten aller p_j geradzahlig wird
  - Also muß für alle $j \leq m$ gelten: $e_{i_1,j} \cdot z_1 + e_{i_2,j} \cdot z_2 .. + e_{i_k,j} \cdot z_k \bmod 2 = 0$
  - Liefert lineares System von m Gleichungen mit k Unbekannten $z_i \in \{0, 1\}$

Nur effizient, wenn alle Primfaktoren der b_i klein sind

Effiziente Bestimmung quadratischer Kongruenzen

- **Einfache Grundidee**
  - Wähle eine Faktorbasis $\mathcal{B} = \{p \text{ prim} \mid p \leq b\}$ für eine Zahl b
  - Zahlen sind effizient zerlegbar, wenn alle Primfaktoren aus $\mathcal{B}$ kommen

    Eine Zahl x heißt ***b*-glatt** wenn kein Primfaktor von x größer als b ist.
  - Suche Zahlen x_i sodaß $b_i = x_i^2 \bmod n$ b-glatt ist

    Suche kann systematisch oder zufallsbasiert sein

- **Kombination von Faktorzerlegungen am Beispiel**
  - Sei $n = 15770708441$ und $\mathcal{B} = \{-1, 2, 3, 5, 7, 11, 13\}$
  - Betrachte $8340934156^2 \bmod n = 21 = 3 * 7$

    $12044942944^2 \bmod n = 78 = 2 * 3 * 13$

    $2773700011^2 \bmod n = 182 = 2 * 7 * 13$
  - Das Produkt der drei Quadrate ergibt $2^2 \cdot 3^2 \cdot 7^2 \cdot 13^2 = 546^2$
  - Ergebnis $x = 9503435785$ und $y = 546$ und $gcd(x-y, n) = 115979$

WIE WÄHLT MAN GUT FAKTORISIERBARE b_i?

- **Erfolglose Probedivisionen sind aufwendig**
  - Probedivision von b durch Elemente von $\mathcal{B}$ benötigt Zeit $\mathcal{O}(|\mathcal{B}|^2 \cdot \|b\|)$
  - Faktorbasen sind i.a. sehr groß (mehr als 100000 Primzahlen)
- **Dixon Random Squares** $\mathcal{O}(e^{(1+o(1))\cdot\|n\|^{1/2}\cdot(\log(\|n\|))^{1/2}})$
  - Probedivision mit semi-zufälliger Wahl der x_i
  - Wahrscheinlichkeit b-glatter Zahlen ist relativ hoch
- **Quadratisches Sieb** $\mathcal{O}(e^{(1+o(1))\cdot\|n\|^{1/2}\cdot(\log(\|n\|))^{1/2}})$
  - Wähle $x_i = \lfloor\sqrt{n}\rfloor + i)$ für $i = 0, \pm 1, \pm 2, \ldots \pm C$
  - Probedivision für $i \in \{0, .., p-1\}$ identifizert alle durch p teilbaren b_i
- **Zahlkörpersieb** $\mathcal{O}(e^{1.92\cdot\|n\|^{1/3}\cdot(\log(\|n\|))^{2/3}})$
  - Systematischer Erzeugung der Kongruenzen $x_i^2 \equiv y_i^2 \bmod n$ mithilfe der algebraischen Zahlentheorie (aufwendig!)
  - Bestes asymptotisches Verhalten aller Faktorisierungsalgorithmen

Quadratische Siebe

- **Wähle Siebintervall $S = \{-C, ... -1, 0, 1, ...C\}$**
  - Für alle $i \in S$ wähle $x_i = \lfloor\sqrt{n}\rfloor + i$ und berechne $b_i := x_i^2 \bmod n$
- **Identifiziere b-glatte Werte im Intervall simultan**
  - Für $p \in \mathcal{B}$ teste alle b_i mit $i \in \{0, .., p-1\}$ auf Teilbarkeit
    $x^2 \bmod p$ ist Polynom zweiten Grades, also gibt es maximal 2 Treffer
  - Ist b_j durch p teilbar, dann auch $b_{j\pm p}, b_{j\pm 2p}, \ldots$ aber kein anderes b_i
  - Es gibt (fast) keine erfolglosen Divisionen mehr
- **Faktorisierung mit quadratische Sieben**
  - Für alle $p \in \mathcal{B}$: Identifiziere die durch p teilbaren b_j mit $j \in \{0, .., p-1\}$
    Dividiere alle $b_{j\pm k\cdot p}$ mit $k \leq C/p$ durch das maximale p^e
  - Ein b_i ist b-glatt, wenn es insgesamt zu 1 oder -1 reduziert wurde
  - Löse Gleichungssystem, wenn $m = |\mathcal{B}|$ b-glatte Zahlen gefunden sind
  - Berechne $x = \prod_{j=1}^{m} x_j^{z_j}$, das zugehörige y und den Faktor $gcd(x-y, n)$

Faktorisierung mit quadratischen Sieben

- **Beispielfaktorisierung von $n = 7429$**
  – Berechne b_i für $S = \{-6, ..., 6\}$ und siebe mit $\mathcal{B} = \{2, 3, 5, 7\}$

i	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
x_i	80	81	82	83	84	85	86	87	88	89	90	91	92
b_i	-1029	-868	-705	-540	-373	-204	-33	140	315	492	671	852	1035
Sieb mit 2		-217		-135		-51		35		123		213	
Sieb mit 3	-343		-235	-5		-17	-11		35	41		71	115
Sieb mit 5			- 47	-1				7	7				23
Sieb mit 7	-1	- 31						1	1				

  – Glatte Werte $b_{-6} = (-1){\cdot}2^2{\cdot}7^3$, $b_{-3} = (-1){\cdot}2^2{\cdot}3^2{\cdot}5$, $b_1 = 2^2{\cdot}5{\cdot}7$ und $b_2 = 3^2{\cdot}5{\cdot}7$
  – Lösung des Gleichungssystems ergibt
  $x = x_1{\cdot}x_2 \bmod n = 227$ und $y = 2{\cdot}3{\cdot}5{\cdot}7 \bmod n = 210$
  – Faktoren sind $gcd(x-y, n) = 17$ und $gcd(x+y, n) = 437$

- **Größe von Siebintervall S und Faktorbasis $\mathcal{B}$**
  – Ideale Größe ist $|\mathcal{B}| \approx 2^{(\|n\|\cdot \log \|n\|)^{1/2}/2}$ und $|S| \approx |\mathcal{B}|{\cdot}u^u$,
  wobei $u = (\|n\|/\log \|n\|)^{1/2}$ (siehe Laufzeitanalyse)

Typische Werte

Bitgröße	128	192	256	384	512
$\|\mathcal{B}\|$ in Tausend	1	4	65	524	16777
$\|S\|$ in Millionen	.1	4	67	536	209715

Laufzeitanalyse quadratischer Siebe

Laufzeit hängt von vielen Faktoren ab

- **Größe eines Intervalls S**
  - Es müssen genügend b-glatte Werte b_i erzeugt werden können
- **Aufwand für Berechnung aller b_i für $i \in S$**
  - Insgesamt $|S|$ Quadrierungen modulo n $\quad \mathcal{O}(|S|{\cdot}\|n\|^2)$
- **Bestimmung der teilbaren b_i mit $i \in \{0, .., p{-}1\}$**
  - Maximal $m^2 = |\mathcal{B}|^2$ Divisionsversuche $\quad \mathcal{O}(|\mathcal{B}|^2{\cdot}\|n\|^2)$
- **Aussieben der b-glatten Elemente in S**
  - Aufwand für eine Divisionen von b_i durch ein $p \in \mathcal{B}$ $\quad \mathcal{O}(\|b_i\|{\cdot}\|p\|)$
  - Zahl der Elemente, die durch ein $p \in \mathcal{B}$ dividiert werden $\quad |S|/p$
  - Anzahl der Elemente der Faktorbasis $\mathcal{B}$ $\quad m$
- **Lösen des linearen $m{\times}m$-Gleichungssystems**
  - Bei dünn besetzten Matrixen (Wiedemann Algorithmus) $\quad \mathcal{O}(m^2{\cdot}\|n\|)$
- **Berechnung von x, y und $gcd(x{-}y, n)$**
  - Jeweils m Multiplikationen bzw. Euklids Algorithmus $\quad \mathcal{O}(m{\cdot}\|n\|^2)$

Laufzeitanalyse quadratischer Siebe (II)

- **Wieviele b-glatte Elemente erzeugt Intervall S?**

  Satz (Zahlentheorie): Sei $\psi(k,b)$ die Anzahl b-glatter Zahlen in $\{1..k\}$.

  Dann ist $\psi(k,b) \approx k/w^w$ (d.h. Anteil $1/w^w$), wobei $w = \|k\|/\|m\|$

  – Quadratische Siebe generieren $x_i = \lfloor\sqrt{n}\rfloor + i$ und $b_i \approx 2i \cdot \lfloor\sqrt{n}\rfloor + i^2$,
  Beide Werte liegen nahe bei $\lfloor\sqrt{n}\rfloor$, da $i \in S$ klein relativ zu $\lfloor\sqrt{n}\rfloor$

  – Anteil b-glatter Werte für S entspricht Anteil in $\{1, .., \lfloor\sqrt{n}\rfloor\}$
  da $\psi(k,b)/k$ sich für kleine Änderungen von k kaum ändert

  – Um m gute b_i zu erzeugen, muß $|S| = m \cdot w^w$ sein mit $w = \|n\|/(2\|m\|)$

- **Aufwand für Aussieben der b-glatten Elemente**

  – Aufwand der Division von b_i durch $p \in \mathcal{B}$ ist $\mathcal{O}(\|b_i\| \cdot \|p\|)$
  mit $b_i \approx \sqrt{n}$ und $p < n$ ist der Aufwand $\mathcal{O}(\|n\|^2/2)$

  – Anzahl der b_i, die durch $p \in \mathcal{B}$ dividiert werden ist $|S|/p$
  Im Mittel ist dies maximal $2 \cdot |S|/m = 2 \cdot w^w$

  – Bei m Elementen in $\mathcal{B}$ ist die Laufzeit insgesamt $\mathcal{O}(m \cdot w^w \cdot \|n\|^2)$

LAUFZEITANALYSE QUADRATISCHER SIEBE (III)

- **Bestimme ideale Größe der Faktorbasis**
  - Gesamtlaufzeit ist $\mathcal{O}(m{\cdot}w^w{\cdot}\|n\|^2 + m^2{\cdot}\|n\|^2)$
  - Polynomieller Anteile verschwinden gegenüber exponentiellem w^w
  - Ergibt Laufzeit $m{\cdot}w^w{\cdot}\|n\|^2 = 2^{\|m\|+w\cdot\log w+2\cdot\log\|n\|}$ mit $w = \|n\|/(2\|m\|)$
  - Der Exponent ist minimal für $\|m\| = 1/2{\cdot}(\|n\|\cdot\log\|n\|)^{1/2}$,
  - Dann ist $w = (\|n\|/\log\|n\|)^{1/2}$ und die Laufzeit ist

    $\mathcal{O}(2^{(\|n\|\cdot\log\|n\|)^{1/2}/2+(\|n\|/\log\|n\|)^{1/2}\cdot(\log\|n\|-\log\log\|n\|)/2+2\cdot\log\|n\|})$

    $= \mathcal{O}(2^{(\|n\|\cdot\log\|n\|)^{1/2}(1-(\log\log\|n\|/\log\|n\|))}) = \mathcal{O}(2^{(1+o(1))\cdot\|n\|^{1/2}\cdot\log\|n\|^{1/2}})$

- **Definiere $L_n[u,v] := \mathcal{O}(e^{v\cdot\|n\|^u\cdot(\log(\|n\|))^{1-u}})$**
  - u beschreibt den Grad der "Exponentialität" der Laufzeitfunktion

    $L_n[0,v] = \mathcal{O}(e^{v(\log(\|n\|))}) = \mathcal{O}(\|n\|^v)$ ist polynomielle Laufzeit

    $L_n[1,v] = \mathcal{O}(e^{v\cdot\|n\|})$ ist exponentielle Laufzeit
  - Schnelle Faktorisierungsalgorithmen sind subexponentiell $(0<u<1)$
  - **Laufzeit quadratischer Siebe ist $L_n[1/2,\ 1+o(1)]$**

Faktorisierungsstrategie für grosse Zahlen

Kaskadischer Einsatz von Verfahren

- **Teste auf kleine Faktoren mit Probedivision**
  – Sehr erfolgreich, wenn n einen Faktor kleiner als 10^7 hat
- **Teste Spezialsituationen mit Pollard $p-1$ /Fermat**
  – Entdeckt Faktoren p, für die $p-1$ nur kleine Primfaktoren hat
  – Entdeckt Faktoren nahe bei $\sqrt{n}$
- **Teste n mit Pollard ρ**
  – Gut, wenn ein Faktor kleiner als 10^{12} ist
- **Teste mit quadratischem Sieb**
  – Empfehlenswert für Zahlen bis ca. 10^{120}
  – Basis und Intervall werden ab 380 Bits zu groß
- **Verwende Zahlenkörpersieb für größere Zahlen**
  – Schnellster bekannter Algorithmus für sehr große Zahlen
  – Einzelne RSA Schlüssel mit 640 Bit wurden erfolgreich faktorisiert

Weitere Angriffe auf RSA

Nicht nur der Schlüssel selbst ist angreifbar

- **Angriff auf kleine Verschlüsselungsexponenten**
  - Verschlüsselung ist sehr effektiv bei kleinen Exponenten
  - Wenn Sender den gleichen Exponenten bei verschiedenen Empfängern nutzt, wird das System leicht angreifbar
  - Beispiel: Nachricht x wurde mit $e = 3$ an drei Empfänger geschickt

    Angreifer liest $y_i = x^3 \bmod n_i$ und löst mit dem Chinesischen Restsatz die Kongruenzen $z \equiv y_i \bmod n_i$ in $\mathbb{Z}_{n_1 \cdot n_2 \cdot n_3}$

    Wegen $x^3 < n_1 \cdot n_1 \cdot n_3$ ist $x = \sqrt[3]{z}$ (ohne Modulararithmetik)
  - Idee läßt sich verallgemeinern auf e Gleichungen für $e < 10^6$
- **Angriff auf kurze Nachrichten**
  - Ist $x^e < n$, so reicht konventionelles Wurzelziehen zur Dechiffrierung
  - Ist $\|x\|$ klein, so ist eine Wörterbuchattacke möglich

Weitere Angriffe auf RSA (II)

- **Homomorphieeigenschaft (Multiplikativität)**
  - $(x_1 \cdot x_2)^e = x_1^e \cdot x_2^e$ macht adaptive chosen ciphertext Attacke möglich
    - Angreifer liest $y \equiv x^e \bmod n$, ergänzt $y' \equiv x'^e \bmod n$ für ein z, schickt $y \cdot y'$ an Empfänger und bittet um Bestätigung
    - Empfänger schickt $m = (y \cdot y')^d$ auf sicherem Kanal zurück
    - Wegen $(y \cdot y')^d = x \cdot x'$ kann $x = m \cdot x'^{-1} \bmod n$ berechnet werden
- **Common Modulus Attacke**
  - Szenario: Zentrale Autorität legt Schlüssel $K := (n, p, q, d_i, e_i)$ für eine gesamte Organisationseinheit fest (gleiches n für alle!)
  - Wenn Angreifer ein einziges Paar (d_i, e_i) in die Hand bekommt, kann n faktorisiert werden und alle Schlüssel liegen offen
  - Alle Mitarbeiter können die für andere bestimmten Nachrichten lesen
- **Cycling Attacke**
  - Ist $y \equiv x^e \bmod n$, so gilt $y^{e^k} \equiv y$ für ein k und $y^{e^{k-1}} \equiv x^{e^k} \equiv x$
  - Berechne $y^e, (y^e)^e, y^{e^3}, \ldots \bmod n$ bis $y^{e^k} \equiv y$ und extrahiere $x = y^{e^{k-1}}$

Wieners Attacke auf Entschlüsselung

- **Angriff auf kleine Entschlüsselungsexponenten**
  - Durchführbar wenn $3d < n^{1/4}$ (also $\|d\| < \|n\|/4-1$) und $q<p<2q$ (Faktoren nahe beieinander, aber zu weit für Fermat)
  - Wegen $e{\cdot}d-1 = k{\cdot}\varphi(n)$ für ein k folgt hieraus $|\frac{e}{n}-\frac{k}{d}| < \frac{1}{3d^2}$ (Beweis folgt)
- **Verwendet zahlentheoretischen Satz**
  - *Ist der Abstand $|\frac{a}{b}-\frac{c}{d}|$ zwischen zwei Brüchen maximal $\frac{1}{2d^2}$, dann ist $\frac{c}{d}$ einer der Konvergenten der Kettenbruchexpansion von $\frac{a}{b}$*
  - Die **Kettenbruchexpansion $[q_1, .., q_m]$ von $\frac{a}{b}$** ist die Folge der Quotienten $q_i = \lfloor\frac{a_i}{b_i}\rfloor$ bei Abarbeitung des Euklidischen Algorithmus
  - Es gilt: $\frac{a}{b} = q_1+\frac{a_1-q_1b_1}{b_1} = q_1+\frac{b_2}{a_2} = q_1+\frac{1}{\frac{a_2}{b_2}} = q_1+\frac{1}{q_2+\frac{1}{\frac{a_3}{b_3}}} = q_1+\frac{1}{q_2+\frac{1}{q_3+\dots\frac{1}{q_m}}}$
  - Der **j-te Konvergent** von $[q_1, .., q_m]$ ist der Kettenbruch $[q_1, .., q_j]$

Liefert Verfahren zur Berechnung von $\frac{k}{d}$

Wieners Attacke im Detail

- **Geringer Abstand der Brüche**
  – Es ist $|\frac{e}{n} - \frac{k}{d}| = |\frac{e \cdot d - k \cdot n}{d \cdot n}| = |\frac{1 + k \cdot (\varphi(n) - n)}{d \cdot n}| < \frac{3k \cdot \sqrt{n}}{d \cdot n} < \frac{3d}{d \cdot \sqrt{n}} = \frac{3}{\sqrt{n}} < \frac{1}{3d^2}$
  – Wegen $q<p<2q$ gilt: $0 < n - \varphi(n) = p+q-1 < 3q < 3\sqrt{n}$
  – Wegen $3d < n^{1/4}$ gilt $\frac{1}{\sqrt{n}} < \frac{1}{9 \cdot d^2}$
- **Bestimmung des j-ten Konvergenten**
  – Der durch $[q_1, .., q_j]$ dargestellte Bruch $\frac{k_j}{d_j}$ ist iterativ zu berechnen:

$$k_j = \begin{cases} 1 & \text{falls } j{=}0 \\ q_1 & \text{falls } j{=}1 \\ q_j k_{j-1} + k_{j-2} & \text{falls } j{\geq}1 \end{cases} \qquad d_j = \begin{cases} 0 & \text{falls } j{=}0 \\ 1 & \text{falls } j{=}1 \\ q_j d_{j-1} + d_{j-2} & \text{falls } j{\geq}1 \end{cases}$$

- **Wieners Angriff auf RSA**
  – In Stufe j der Attacke berechne den j-ten Konvergenten $\frac{k_j}{d_j}$
  – Setze $\varphi_j = (e \cdot d_j - 1)/k_j$ und löse Gleichung $p^2 - (n - \varphi_j + 1) \cdot p + n = 0$
  – Wenn beide Lösungen zwischen 2 und n liegen, sind sie Teiler von n und d_j ist der Entschlüsselungsschlüssel

Beispielattacke nach Wieners Methode

- **Gegeben $n = 160523347$ und $e = 60728973$**
  - Zahlenbruchentwicklung von $\frac{e}{n}$: $[0, 2, 1, 1, 1, 4, 12, 102, 1, 1, 2, 3, 2, 2, 36]$
  - Konvergenten sind $\frac{0}{1}, \frac{1}{2}, \frac{1}{3}, \frac{2}{5}, \frac{3}{8}, \frac{14}{37}, \frac{171}{452}, \ldots$
  - Erste fünf Konvergenten liefern keine Faktorisierung von n
- **Berechnung der sechsten Stufe der Attacke**
  - φ_6 ist $(37{\cdot}60728973 - 1)/14 = 160498000$
  - Zu lösende quadratische Gleichung: $p^2 - 25348{\cdot}p + 160523347 = 0$
    ergibt $q = 12347$ und $p = 13001$
- **Ergebnis der Attacke**
  - Faktorisierung von $\boldsymbol{n = 12347{\cdot}13001}$
  - Entschlüsselungsschlüssel $\boldsymbol{d = 37}$ (knapp unter $n^{1/4}/3 = 37.52004$)

Richtlinien für die Schlüsselauswahl

- **p und q müssen sehr groß sein**
  - Nach heutigen Maßstäben sind mindestens 512 Bit erforderlich
  - Für sicherheitskritische Anwendungen sind 2024 Bit empfehlenswert
  - $\|p\|$ und $\|q\|$ sollten ähnlich groß sein (nur wenige Bits Unterschied)
- **Zufällige Primzahlen generieren**
  - Systematische Konstruktion kann nachgebaut werden
  - Es ist besser, Eigenschaften im Nachinein zu prüfen
- **Starke Primzahlen auswählen**
  - p und q dürfen nicht zu nahe beieinanderliegen (Fermat-Faktorisierung!)
  - $p{-}1$ muß auch große Primfaktoren haben (Pollard $p{-}1$ Faktorisierung!)
  - $p{+}1$ muß auch große Primfaktoren haben (Williams $p{+}1$ Faktorisierung!)
- **e und d müssen groß sein**
  - Vermeide Angriffe auf zu kleine Ver-/Entschlüsselungsexponenten