Kryptographie I

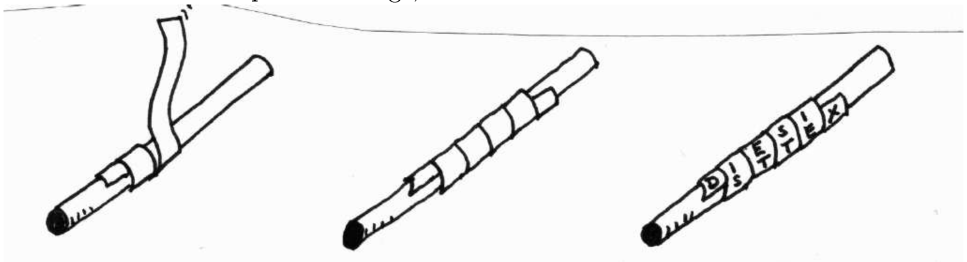
Dieser Text ist eine Zusammenfassung zweier Vorträge bei Wochenendseminaren der Leipziger Schülergesellschaft Mathematik. Teile dieses Skriptes wurden von Heinrich-Gregor Zirnstein ausgearbeitet und geTeXt, wofür ich mich herzlich bedanken möchte.

1 Was ist Kryptographie?

Kurz gesagt: Kryptographie ist die Wissenschaft vom Verschlüsseln und Entschlüsseln.
Das Wort "Kryptographie" kommt aus dem Altgriechischen. Dort bedeutet:

$$\kappa\rho\upsilon\pi\tau o\varsigma \text{ — versteckt}$$
$$\gamma\rho\alpha\varphi\varepsilon\iota\nu \text{ — schreiben}$$

Aber nicht nur das Wort ist alt, sondern auch die Kryptographie selbst: Denn Menschen wollten schon immer Nachrichten vor anderen geheimhalten.
Ein schönes Beispiel ist die *Skytala* (Stab). Diese wurde um 500 v.Chr. von den Spartanern im alten Griechenland verwendet.
Das Verfahren ist einfach: Man wählt einen Stab, wickelt einen Papierstreifen mehrfach darum herum und schreibt dann den Text auf den Streifen, so daß jeder Buchstabe auf einer neuen Papierbahn liegt, etwa so:



Der abgerollte Papierstreifen zeigt dann eine ziemlich wirr aussehende Buchstabenfolge, die aber durch Aufrollen auf einen Stab gleicher Größe wieder gelesen werden kann.
Ein anderes historisches Verfahren wird von Caesar berichtet: Er ersetzte in seinen Briefen an Cicero jedes "A" durch ein "D", jedes "B" durch ein "E" und so fort.
Heute spielt Kryptographie in vielen Bereichen eine wichtige Rolle – nicht nur in Geheimdiensten, an die man dabei vielleicht zuerst denken würde! So werden zum Beispiel für das Homebanking im Internet Daten verschlüsselt, damit nicht Unbefugte das System für Ihre Zwecke mißbrauchen können (indem sie vielleicht fremdes Geld auf Ihr Konto buchen). Auch schnurlose Telefone und Handys benutzen Verschlüsselungen, damit niemand unerlaubt mithören kann. Kurz gesagt findet moderne Kryptographie immer da ihren Einsatz, wo wichtige Daten übermittelt werden. Und hinter moderner Kryptographie steckt (auch wenn man das vielleicht nicht gleich erwarten würde) eine Menge Mathematik. Und damit wollen wir uns in diesem Kurs beschäftigen.

2 Kurze Einführung in die Theorie

2.1 Kryptographische Begriffe

Zunächst mal müssen wir unsere vage Vorstellung von Kryptographie etwas präzisieren. – Schließlich wollen wir am Ende Mathematik einsetzen, und brauchen dafür präzise formulierte Probleme.
Wir unterscheiden dafür zwischen dem Verschlüsselungs**verfahren** und dem **Schlüssel**

(auch "Code" genannt). Nur den Schlüssel können wir geheimhalten, das Verschlüsselungsverfahren nicht! (Das nennt man "Kerckhoffsches Prinzip".)
In unseren zwei historischen Beispielen wäre der Schlüssel also einmal der Stab (die Skytala), um den unser Text zu wickeln ist. Im Falle Caesars wäre der Schlüssel die Zahl drei, denn jeder Buchstabe wird ja um drei Buchstaben im Alphabet weiter verschoben.
Natürlich kann man in beiden Fällen durch Ausprobieren die Schlüssel sehr leicht herausfinden. Offenbar hofften die Benutzer darauf, daß andere ihre Verfahren nicht kannten. – Daß dies heute, wo an einem Verschlüsselungsprogramm vielleicht hundert Programmierer arbeiten, nicht mehr gut gehen würde, liegt auf der Hand! Dies ist eben auch der Grund für das Kerckhoffsche Prinzip.
Wir wollen nun den Verschlüsselungsvorgang mathematisch beschreiben. Eine Nachricht (auch Klartext genannt) fassen wir dabei als natürliche Zahl auf. Wenn wir stattdessen einen Text verschlüsseln wollen, so können wir diesen ja durch eine geeignete Übertragung immer in Zahlen umwandeln. In Computern wird dazu meist das ASCII-System verwendet. Der Großbuchstabe "A" bekommt dabei z.B. die Zahl 64 zugeordnet, die Leerstelle die Zahl 32.
Eine Verschlüsselung ist nun die Anwendung einer Abbildung f auf eine Nachricht:

Definition 1 *Sei m die Nachricht (Klartext), so ist $c = f_k(m)$ der* ***verschlüsselte Text****, f die* ***Verschlüsselungsfunktion*** *und k der* ***Schlüssel****. f muß dabei* ***umkehrbar*** *sein, die Umkehrung nennen wir f_k^{-1}, d.h. es muß für jede natürliche Zahl m gelten: $f_k^{-1}(f_k(m)) = m$.*

Die Umkehrbarkeitsbedingung sagt einfach, daß wir aus dem verschlüsselten Text auch tatsächlich wieder den Ursprungstext erhalten können! Wenn wir als f_k zum Beispiel $f_k(m) = (m-2)^2$ wählen würden, dann hätten die Nachrichten $m = 1$ und $m = 3$ denselben verschlüsselten Text, denn $f_k(1) = (1-2)^2 = 1$ und $f_k(3) = (3-2)^2 = 1$. Das wäre natürlich keine sinnvolle Verschlüsselung, da dabei Information verlorenginge! Die Bedingung ist also eigentlich selbstverständlich.
Wir haben nun definiert, was eine Verschlüsselung ist. Nach dieser Definition wäre aber zum Beispiel $f_k(m) := m$ ebenfalls eine Verschlüsselungsfunktion. Diese Funktion ist natürlich keine brauchbare Verschlüsselungsfunktion! Eine solche Funktion soll eben auch *sicher* sein, d.h. man soll ohne Kenntnis des Schlüssels k nicht so ohne weiteres die Nachricht herausfinden können.
Deshalb definieren wir:

Definition 2 *Eine Verschlüsselungsfunktion f_k ist* ***sicher****, wenn m ohne Kenntnis von k aus $f_k(m)$ nicht (oder in der Praxis nicht) bestimmt werden kann, auch wenn einer der folgenden Fälle vorliegt:*

- *"Verschlüsselter-Text-Angriff": Es sind außerdem noch einige verschlüsselte Texte $f_k(m_1), \ldots, f_k(m_n)$ zum selben Schlüssel k bekannt.*
- *"Bekannter-Klartext-Angriff": Es sind einige Klartexte mit verschlüsseltem Text bekannt, also $m_1, \ldots m_n$ und $f_k(m_1), \ldots, f_k(m_n)$.*
- *"Wählbarer-Klartext-Angriff": Es ist f_k bekannt und damit auch $f_k(m)$ für beliebige Texte m.*
- *"Wählbarer-verschlüsselter-Text-Angriff": Es ist sogar f_k^{-1} bekannt, nur eben k nicht.*

Das sind ziemlich harte Bedingungen! Es fragt sich, ob wir ein Verfahren finden werden, das ihnen genügt. Außerdem müssen wir noch definieren, was die Formulierung "in der Praxis" bedeuten soll. – Kann man nicht mit genügend Versuchen

jeden Code knacken? Und hat man dafür nicht Computer, die immer schneller werden?
Im nächsten Abschnitt werden wir lernen, daß die Sache so einfach nicht ist – glücklicherweise!

2.2 Landau-Symbole

Was bedeutet, daß etwas "in der Praxis" nicht berechnet werden kann? Grob gesagt bedeutet das, daß der Rechenaufwand dafür dermaßen groß ist, daß das Problem selbst mit viel besseren Computern als heute vorhanden nicht gelöst werden könnte. Aber wie stellt man das sicher? Nun, dazu brauchen wir eine mathematische Definition:

Definition 3 (Landau-Symbole) *Wir sagen $f(n) = \mathcal{O}(g(n))$ für $n \to \infty$, wenn gilt: $\frac{f(x)}{g(x)} \to c \neq 0$ für $n \to \infty$.*

Zwei Beispiele dazu:
Sei $f(n) := n^3 - n^2$, dann ist $f(n) = \mathcal{O}(n^3)$ für $n \to \infty$, denn:

$$\frac{f(n)}{n^3} = \frac{n^3 - n^2}{n^3} = \frac{n^3}{n^3} - \frac{n^2}{n^3} = 1 - \frac{1}{n},$$

und da $\frac{1}{n}$ der Null immer näher kommt, wenn n immer größer wird, d.h. $\frac{1}{n} \to 0$ für $n \to \infty$, ist die obige Definition erfüllt.
Ein zweites Beispiel:
Sei $f(n) := 3 \cdot 2^n + n^3$, dann ist $f(n) = \mathcal{O}(2^n)$. Ein Beweis würde hier jetzt zu weit führen, aber wenn ihr $\frac{f(n)}{2^n}$ auf dem Computer plottet, dann könnt Ihr zumindest einen Eindruck davon bekommen, daß sich dieser Wert für $n \to \infty$ der Zahl 3 annähert. (Normalerweise sollte man sowas beweisen, denn manchmal führen solche Computergrapfiken auch in die Irre, wie ich auch schon mal feststellen mußte!)
Soviel zum Landau-Symbol $\mathcal{O}()$. Nun wenden wir diese Definition an.
Wir nehmen an, daß wir zur Berechnung von $f_k(m)$ genau s_k Rechenschritte benötigen, zur Entschlüsselung (ohne Kenntnis von k) jedoch $d(k)$ Rechenschritte.
Wenn $d(k) = 2s_k$, dann bräuchten wir zum Entschlüsseln nur doppelt solange wie zum Verschlüsseln, und das wäre natürlich schlecht für unser Verfahren, es wäre eben *unsicher*. Allgemein ist der Fall ungünstig, in dem $d(k) = \mathcal{O}(s_k)$ ist, da wir dann nur einen festen Faktor c länger zum Entschlüsseln brauchen als zum Verschlüsseln verwendet wurde.
Besser sieht die Situation aus, wenn $d(k) = \mathcal{O}(10^{s_k})$ ist, wie folgende Tabelle verdeutlicht, in der der Einfachheit halber sogar $d(k) = 10^{s_k}$ ist:

s_k	$d(k)$
1	10
2	100
3	1000
...	...
10	10000000000
...	...
100	10000000000000000000000000000000000000000000000000000000 0000000000000000000000000000000000000000000000000000000

Wenn wir nur 100 Rechenschritte zum Verschlüsseln verwenden, haben wir also schon viel zu lange mit dem Entschlüsseln (ohne Schlüssel) zu tun! Und wir können selbst bei kurzen Verschlüsselungszeiten erreichen, daß die Entschlüsselungszeiten beliebig hoch werden.
Warum helfen da schnellere Computer nicht? Nun, schnellere Computer verringern

den Zeitaufwand immer nur um einen bestimmten Faktor. Aber durch eine nur unmerklich längere Verschlüsselungszeit läßt sich dieser Vorteil wieder mehr als wettmachen. – Das kannst Du Dir übrigens einfach an obiger Tabelle klarmachen: Um wieviel muß man s_k erhöhen, um einen Faktor 100 wieder wettzumachen? Richtig: Nur um 2!
Zusammenfassend gesagt kann man, wenn man eine Methode mit dieser Eigenschaft gefunden hat, in kurzer Zeit ver- und entschlüsseln, wenn man den Schlüssel hat, aber der verschlüsselte Text kann in der Praxis nicht ohne Kenntnis des Schlüssels wieder entschlüsselt werden!
Dies ist die Kernidee moderner Kryptographie. Wir werden später Verfahren kennenlernen, die dies leisten. Sie basieren darauf, daß es sehr lange dauert, große Zahlen in ihre Primfaktoren zu zerlegen. Man vermutet, daß die Primfaktorzerlegung einer Zahl mit k Ziffern die Rechenzeit $\mathcal{O}(10^k)$ benötigt. Ein mathematischer Beweis für diese Vermutung steht jedoch noch aus. Sie ist eine der "Milleniumsprobleme", für deren Lösung ein Preisgeld von einer Million US-Dollar ausgeschrieben ist. Die Multiplikation zweier Zahlen mit etwas $\frac{k}{2}$ Ziffern zu einer Zahl mit k Ziffern (also die Umkehrung der Primfaktorzerlegung) benötigt übrigens nur $\mathcal{O}(k^2)$ Schritte.

2.3 Symmetrische und asymmetrische Verschlüsselung

Bei der **symmetrischen** Verschlüsselung benutzen Sender und Empfänger den gleichen Schlüssel k. Der Sender kennt insbesondere f_k, der Empfänger f_k^{-1}.
Ein Problem bei symmetrischer Verschlüsselung ist, daß sich beide Personen vorher (geheim!) auf einen Schlüssel einigen müssen. Das ist häufig umständlich. Man denke nur an das Beispiel einer Internet-Bank, mit der die Kunden erstmal geheime Schlüssel austauschen sollten. Das wäre nicht nur umständlich, sondern auch ein Sicherheitsrisiko!
Daher gibt es seit einiger Zeit **asymmetrische** Verschlüsselung: Dabei läßt sich die Umkehrfunktion für einen festen Schlüssel berechnen, ohne diesen Schlüssel zu kennen. Man wählt dabei $f_s^{-1}(m) = f_e(m)$. Der Sender benutzt den Schlüssel s. Der Empfänger verschlüsselt die Nachricht nocheinmal mit seinem Schlüssel e, den nur er kennt. Das Ergebnis ist wieder die Nachricht, denn $f_e(f_s(m)) = f_s^{-1}(f_s(m)) = m$. Der Empfänger kann den Schlüssel s öffentlich bekannt machen (man nennt ihn daher auch "öffentlicher Schlüssel"), so daß andere ihm eine entsprechend verschlüsselte Nachricht schicken können. Diese kann nur er (mit seinem Schlüssel e) wieder entschlüsseln und dann lesen.
Unser Ziel wird es sein, ein solches System zu realisieren!

3 Der RSA-Algorithmus

3.1 Modulo-Rechnen

Für den Verschlüsselungsalgorithmus sind Ideen aus der Zahlentheorie unabdingbar, vor allem die Modulo-Rechnung spielt eine wichtige Rolle. Als einführendes Beispiel schauen wir eben auf unsere Uhr und werden vielleicht feststellen, daß es gerade 17.00 Uhr ist. Nun fragen wir uns, wie spät es in 10 Stunden sein wird, und rechnen $17 + 10 = 27$ Uhr. In zehn Stunden wird unsere Uhr aber die Ziffer 3 anzeigen. Diesen Sachverhalt werden wir ab sofort als $27 \equiv 3 \pmod{24}$ (sprich: 27 kongruent 3 modulo 24) schreiben.
Allgemein definiert schreibt man

$$a \equiv b \pmod{c},$$

um auszudrücken, daß a und b denselben Rest bei der Teilung durch c lassen. Das heißt also $a = b + nc$ mit einer natürlichen Zahl n oder wir können auch $c|(a-b)$

schreiben. Dabei soll b die kleinst Der Rest von a bei der Division durch c wird in ähnlicher Weise mit $a \bmod c$ bezeichnet.

Übungsaufgaben 3.1

1. Berechne $(14 + 9) \bmod 5$.
2. Berechne $(13 - 5) \bmod 3$.
3. Berechne $220 \cdot 8 \bmod 5$.
4. Beweise, daß stets $(a \bmod n) \cdot (b \bmod n) = ab \bmod n$ gilt!

3.2 Die Eulersche φ-Funktion

Neben der Modulo-Rechnung ist auch die Eulersche φ-Funktion von Bedeutung. Sie ist nach dem bedeutenden Schweizer Mathematiker Leonard Euler (1707-1783) benannt. Für das weitere Verständis ist dieser Abschnitt nicht zwingend notwendig, er und der Abschnitt 3.3 werden an andere Stelle als Spezialfall auftreten. Wer auf diesen mathematischen Exkurs verzichten möchte, kann im Abschnitt 3.4 weiterlesen.
Die Eulersche φ-Funktion wird folgendermaßen definiert:

Definition 4 *Sei m eine natürliche Zahl. Dann ist $\varphi(m)$ die Anzahl der Zahlen a mit $1 \leq a < m$, für die* $\mathrm{ggT}\,(a, m) = 1$ *ist.*

Als Beispiel wählen wir $m = 6$. Nur die 1 und die 5 sind kleiner als 6 und teilerfremd zur 6, also ist $\varphi(6) = 2$.

Übungsaufgaben 3.2

1. Berechne $\varphi(2)$.
2. Berechne $\varphi(4)$.
3. Berechne $\varphi(p)$ für jede beliebige Primzahl p.

3.3 Der Eulersche Satz

Wozu soll die Eulersche φ-Funktion eigentlich gut sein? Der Eulersche Satz gibt uns die Antwort:

Satz 1 (Eulerscher Satz) *Seien $n, m \in \mathbb{N}$ und* $\mathrm{ggT}\,(m, n) = 1$. *Dann gilt*

$$m^{\varphi(n)} \equiv 1 \pmod{n}.$$

Auf einen Beweis dieses Satzes wird hier verzichtet, aber er findet sich in den meisten Büchern über Algebra. In den folgenden Übungsaufgaben sind jedoch einige Spezialfälle mit Hilfe der Übungsaufgaben aus dem vorangegangenen Abschnitt zu beweisen.

Übungsaufgaben 3.3

1. Beweise den Satz für $n = 4$.
2. Beweise den Satz für $n = 7$.

3.4 Das RSA-Verfahren

Nach der theoretischen Vorarbeit können wir uns nun dem RSA-Verfahren zuwenden. Es wurde nach seinen drei Erfindern benannt: R wie Rivest, S wie Shamir und A wie Adleman, die es zuerst in "Comm. of the ACM 21 (1978), S.120-126" veröffentlichten.
Zur Anwendung des Algorithmus müssen wir zunächst zwei natürliche Zahlen s und d finden, deren Produkt sd wir als

$$sd = k(p-1)(q-1)+1$$

darstellen können. Hierbei sollen p und q verschiedene Primzahlen und k eine natürliche Zahl sein. Um das Knacken dieser Verschlüsselung extrem schwer zu machen, ist man bestrebt, die Primzahlen p und q möglichst groß zu wählen.
Nun brauchen wir noch eine Nachricht, dargestellt durch eine natürliche Zahl m. Jetzt sind wir in der Lage, unsere Verschlüsselungsfunktion f_s zu definieren:

$$f_s(m) := m^s \bmod pq.$$

Als Beispiel werden wir eine kleine Verschlüsselung durchführen. Wir wählen dafür $p = 11$, $q = 5$ und $k = 2$ und noch $s = 3$ und $d = 27$. Eine Kontrolle liefert $sd = 81 = 2(11-1)(5-1)+1 = k(p-1)(q-1)+1$, wir haben uns also nicht verwählt. Die Verschlüsselungsfunktion lautet dann

$$f_3 = m^3 \bmod (5 \cdot 11 = 55).$$

Als kurze Übung kann der Leser ein paar Nachrichten verschlüsseln:

Übungsaufgaben 3.4

1. Berechne $f_3(1)$, oder anders gesagt: Verschlüssle die Zahl 1 mit der eben berechneten Verschlüsselungsfunktion!
2. Verschlüssle die 4 (d.h. berechne $f_3(4)$).

Nun haben wir zwar eine Verschlüsselungsfunktion, aber wie wollen wir eine Nachricht wieder entschlüsseln? Die dazu benötigte Entschlüsselungsfunktion f_s^{-1} lautet

$$f_s^{-1} = f_d,$$

was wir später noch beweisen werden.
Als Beispiel entschlüsseln wir die "1":

$$f_{27}(1) = 1^{27} \bmod 55 = 1,$$

in Übereinstimmung mit dem Resultat aus Übung 3.4.1. Nun prüfen wir noch die Nachricht "9":

$$\begin{aligned} f_{27}(9) &= 9^{27} \bmod 55 = (9^2)^{13} \cdot 9 \bmod 55 = 26^{13} \cdot 9 \bmod 55 \\ &= (26^2)^6 \cdot 26 \cdot 9 \bmod 55 = 16^6 \cdot 14 \bmod 55 = (16^2)^3 \cdot 14 \bmod 55 \\ &= 36^3 \cdot 14 \bmod 55 = 16 \cdot 14 \bmod 55 = 4 \end{aligned}$$

Dieses Resultat ist gemäß der Übung 3.4.2. zu erwarten gewesen. Allerdings hätten wir die Rechnung noch wesentlich effizienter gestalten können, wenn wir mit dem Chinesischen Restklassensatz arbeiten würden.

Mit dem RSA-Algorithmus haben wir nun ein **asymmetrisches** Verschlüsselungsverfahren, Empfänger und Sender haben unterschiedliche Schlüssel:

$$\boxed{m}$$
$$\downarrow$$

Verschlüsselung: $f_s(m) = m^s \bmod pq$

$$\downarrow$$

Entschlüsselung: $(f_s(m))^d \bmod pq = f_d(f_s(m)) = f_s^{-1}(f_s(m)) = m$

Nun müssen wir noch beweisen, daß die Entschlüsselungsfunktion wirklich

$$f_s^{-1} = f_d.$$

lautet, und zwar nicht nur für die Spezialfälle, die wir gerade ausgerechnet haben. Dazu berechnen wir $f_d(f_s(m))$, was m ergeben sollte:

$$\begin{aligned} f_d(f_s(m)) &= (f_s(m))^d \bmod pq = (m^s)^d \bmod pq = m^{sd} \bmod pq \\ &= m^{k(p-1)(q-1)+1} \bmod pq \end{aligned}$$

Nun benutzen wir den kleinen Satz von Fermat, benannt nach dem französischen Mathematiker Pierre de Fermat (1610-1665):

Satz 2 (Kleiner Satz von Fermat) *Für jede Primzahl p und jedes natürliche m mit $1 \le m < p$ gilt*

$$m^{p-1} \equiv 1 \pmod p.$$

Wir werden ihn im Abschnitt 3.5 beweisen.
Er sagt uns nun, daß

$$\begin{aligned} m^{p-1} &\equiv 1 \pmod p \text{ und} \\ m^{q-1} &\equiv 1 \pmod q \end{aligned}$$

gilt. Daher ist

$$\begin{aligned} m^{k(p-1)(q-1)} &= (m^{p-1})^{k(q-1)} \equiv 1 \pmod p \text{ und} \\ m^{k(p-1)(q-1)} &= (m^{q-1})^{k(p-1)} \equiv 1 \pmod q. \end{aligned}$$

Das bedeutet, das wir den Rest 1 erhalten, wenn wir $m^{k(p-1)(q-1)}$ durch p teilen. Genauso bleibt der Rest 1, wenn wir durch q teilen. Deshalb muß dann auch bei der Division durch $p \cdot q$ der Rest 1 bleiben:

$$m^{k(p-1)(q-1)} \equiv 1 \pmod{pq}.$$

Für $f_d(f_s(m)) = m^{k(p-1)(q-1)+1} \pmod{pq}$ heißt das dann:

$$\begin{aligned} f_d(f_s(m)) &= m^{k(p-1)(q-1)+1} \bmod pq \\ &= m \cdot m^{k(p-1)(q-1)} \bmod pq = m \cdot 1 \bmod pq \\ &= m \text{ , da } m \text{ kleiner als } pq \text{ ist.} \end{aligned}$$

Also ist tatsächlich $f_d(f_s(m)) = m$. Das besagt nichts anderes, als daß $f_d = f_s^{-1}$ die Entschlüsselungsfunktion ist: Wenn wir die Nachricht m erst verschlüsseln (mit f_s) und anschließend entschlüsseln (mit f_d), dann erhalten wir wieder die Ausgangsnachricht m.
Abschließend bleibt noch zu klären, warum dieses Verfahren so sicher ist. Wir erinnern uns, daß wir aus der Kenntnis des Schlüssels s und der beiden Primzahlen p und q den zur Entschlüsselung benötigten Schlüssel d berechnen können. Wir kennen das Produkt pq, wir brauchen es ja zum Verschlüsseln. Das Problem besteht nun darin, aus pq die beiden Zahlen p und q zu berechnen. Das dauert aber sehr lange, und wenn wir die Zahlen sehr groß wählen, ist es praktisch unmöglich p und q zu berechnen, wie wir schon im Abschnitt 2.2 über Landau-Symbole erwähnt haben.

3.5 Beweis des kleinen Satzes von Fermat

Wir wollen in diesem Abschnitt den schon verwendeten kleinen Satz von Fermat beweisen. Dazu benötigen wir zwei Lemmata:

Lemma 1 *Sei p eine Primzahl. Dann ist* $\left(\begin{array}{c} p \\ i \end{array}\right) = \frac{p!}{i!(p-i)!}$ *durch p teilbar. Dabei soll i eine natürliche Zahl mit $1 < i < p$ sein.*

Beweis. Nehmen wir an, p teilt nicht $\left(\begin{array}{c} p \\ i \end{array}\right) = \frac{p!}{i!(p-i)!}$. Dann müßte der Nenner des Bruches durch p teilbar sein, ansonsten würde der Faktor p im Zähler $p!$ nicht "herausgekürzt" werden. Wenn wir aber den Nenner ausschreiben, erhalten wir

$$1 \cdot 2 \cdot \dots \cdot i \; \cdot \; 1 \cdot 2 \cdot \dots \cdot (p-i).$$

Wir sehen, daß p gar nicht als Faktor in diesem Produkt auftaucht, wenn wie vorausgesetzt $1 < i < p$ ist. Demnach müßte p zusammengesetzt sein, im Widerspruch dazu, daß p eine Primzahl ist. q.e.d.

Nun können wir das zweite Lemma beweisen:

Lemma 2 *Für jede Primzahl p und zwei beliebige natürliche Zahlen a, b gilt stets*

$$(a+b)^p \equiv a^p + b^p \pmod{p}$$

Beweis. Wir berechnen $(a+b)^p$ mit Hilfe des Binomischen Lehrsatzes:

$$(a+b)^p = a^p + \left(\begin{array}{c} p \\ 1 \end{array}\right) a^{p-1}b + \dots + \left(\begin{array}{c} p \\ p-1 \end{array}\right) ab^{p-1} + b^p.$$

Da nach Lemma 1 die Summanden mit den Binominalkoeffizienten alle durch p teilbar sind, gilt

$$a^p + \left(\begin{array}{c} p \\ 1 \end{array}\right) a^{p-1}b + \dots + \left(\begin{array}{c} p \\ p-1 \end{array}\right) ab^{p-1} + b^p \equiv a^p + b^p \pmod{p},$$

und somit gilt

$$(a+b)^p \equiv a^p + b^p \pmod{p}.$$

q.e.d.

Nun können wir den kleinen Satz von Fermat durch vollständige Induktion über m beweisen:

Induktionsanfang: Für $m = 1$ gilt natürlich $1^{p-1} \equiv 1 \pmod{p}$.

Induktionsschritt: Für $m = m_0$ gelte $m_0^{p-1} \equiv 1 \pmod{p}$. Wir können dafür auch $m_0^p \equiv m_0 \pmod{p}$ schreiben.

Für $m = m_0 + 1$ gilt dann

$$(m_0+1)^p \equiv m_0^p + 1 \equiv m_0 + 1 \pmod{p}.$$

Also haben wir $(m_0+1)^p \equiv m_0 + 1 \pmod{p}$ bzw. $(m_0+1)^{p-1} \equiv 1 \pmod{p}$, wenn nach Voraussetzung $m_0 + 1 \neq p$ ist. q.e.d.

Übungsaufgaben 3.5

> Man beweise den kleinen Satz von Fermat auf eine zweite Weise, und zwar mit Hilfe der Eulerschen φ-Funktion. Als Anleitung sei gesagt, daß man zuerst die Übungsaufgabe 3.2.3 lösen und dann den Eulerschen Satz verwenden sollte.

A Lösung der Übungsaufgaben

• *Übungsaufgabe 1 aus Abschnitt 3.1*
$14 + 9 \bmod 5 = 3$.

• *Übungsaufgabe 2 aus Abschnitt 3.1*
$13 - 5 \bmod 3 = 2$.

• *Übungsaufgabe 3 aus Abschnitt 3.1*
$220 \cdot 8 \bmod 5 = 0$.

• *Übungsaufgabe 4 aus Abschnitt 3.1*
Wir können a als $a = m_a \cdot n + (a \bmod n)$ und b als $b = m_b \cdot n + (b \bmod n)$ darstellen. Für ab erhalten wir die Darstellung.

$$\begin{aligned} ab &= (m_a \cdot n + (a \bmod n)) \cdot (m_b \cdot n + (b \bmod n)) \\ &= n \cdot (m_a \cdot m_b + m_a \cdot (b \bmod n) + m_b \cdot (a \bmod n)) + (a \bmod n)) \cdot (b \bmod n)). \end{aligned}$$

Bei der Teilung von ab durch n bleibt der Rest $ab \bmod n$, der aber, wie wir an der obigen Gleichung ablesen können, gleich $(a \bmod n)) \cdot (b \bmod n))$ ist. q.e.d.

• *Übungsaufgabe 1 aus Abschnitt 3.2*
$\varphi(2) = 1$.

• *Übungsaufgabe 2 aus Abschnitt 3.2*
Die zu 4 teilerfremden Zahlen kleiner als 4 sind 1 und 3, also ist $\varphi(4) = 2$

• *Übungsaufgabe 3 aus Abschnitt 3.2*
Jede natürlichen Zahl n, die kleiner als die Primzahl p ist, hat mit p nur den gemeinsamen Teiler 1. Hätte n noch einen Teiler mit p gemeinsam, so müßte n durch p teilbar sein, im Widerspruch dazu, daß $n < p$. Also sind alle $p - 1$ Zahlen n mit $n < p$ zu p teilerfremd und somit gilt

$$\varphi(p) = p - 1$$

• *Übungsaufgabe 1 aus Abschnitt 3.3*
Es ist $\varphi(4) = 2$ und so haben wir

$$a^2 \equiv 1 \pmod 4$$

zu beweisen. Dazu betrachten wir die Reste der Zahl a bei der Division durch 4 und machen die folgende Fallunterscheidung:

$a \bmod 4$	
0	Brauchen wir nicht weiter zu betrachten, da ggT $(a, 4) = 4$.
1	Es ist $a^2 \equiv 1 \pmod 4$ wegen der Übungsaufgabe 4 aus Abschnitt 3.1.
2	Brauchen wir nicht weiter zu betrachten, da ggT $(a, 4) = 2$.
3	Es ist $a^2 \equiv 3^2 \equiv 1 \pmod 4$ wegen der Übungsaufgabe 4 aus Abschnitt 3.1.

Wir haben also für alle möglichen Reste modulo 4 gezeigt, daß der Eulersche Satz für $n = 4$ gilt.

• *Übungsaufgabe 2 aus Abschnitt 3.3*
Da 7 eine Primzahl ist, folgt das Resultat unter Zuhilfenahme der Übungsaufgabe 3 aus Abschnitt 3.2 aus dem kleinen Satz von Fermat. Wir könnten jedoch auch wie in der vorherigen Aufgabe vorgehen und alle möglichen Reste modulo 7 betrachten. Aus Platzgründen wird hier darauf verzichtet.

• *Übungsaufgabe 1 aus Abschnitt 3.4*
$f_3(1) = 1^3 \bmod 55 = 1$.

• *Übungsaufgabe 2 aus Abschnitt 3.4*
$f_3(4) = 4^3 \bmod 55 = 64 \bmod 55 = 9$.

• *Übungsaufgabe aus Abschnitt 3.5*
Nach der Übungsaufgabe 3 aus Abschnitt 3.2 ist $\varphi(p) = p - 1$ für jede Primzahl p. Eingesetzt in den Eulerschen Satz heißt das

$$a^{\varphi(p)} = a^{p-1} \equiv 1 \pmod{p}.$$

Da steht's.

Marc Oliver Rieger