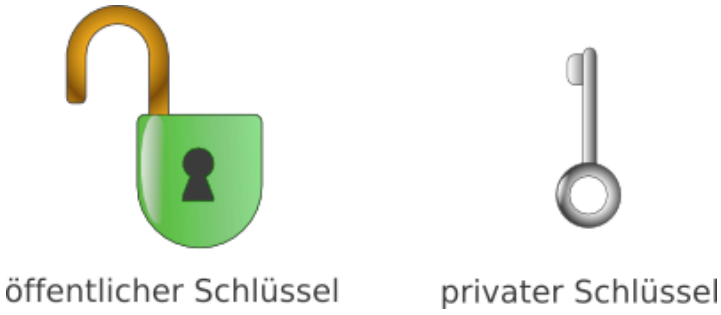
1. Asymmetrische Verschlüsselung einfach erklärt

Das Prinzip der asymmetrischen Verschlüsselung beruht im Wesentlichen darauf, dass sich jeder Kommunikationspartner jeweils ein Schlüsselpaar (bestehend aus zwei Schlüsseln) erzeugt. Einer der Schlüssel wird geheim gehalten, das ist der so genannte **private Schlüssel.** Der zweite Schlüssel wird jedem kommunikationswilligen Teilnehmer zugänglich gemacht. Der zweite Schlüssel heißt deshalb **öffentlicher Schlüssel.**
Der große Vorteil dieses Verfahrens im Vergleich zur symmetrischen Verschlüsselung[1] ist in der einfachen Verteilung des öffentlichen Schlüssels begründet. Dieser kann wirklich für jeden Menschen frei zugänglich sein, ohne dass dadurch das Verfahren unsicher wird.

Das Verständnis asymmetrische Verschlüsselungsverfahren setzt einige mathematische Kenntnisse voraus (siehe Abschnitt 2). Das folgende Beispiel kommt jedoch ganz ohne Mathematik aus. Zum besseren Verständnis wird hier der **öffentliche Schlüssel** durch ein *Vorhängeschloss* symbolisieret und der **private Schlüssel** durch den *passenden Schlüssel für dieses Schloss*.



Nehmen wir nun an, Bob möchte eine Nachricht an Alice schicken. Alice will aber verhindern, dass ihr Vater lesen kann, was Bob und andere Freunde so alles schreiben. Dazu wird sie also als Erstes einige Schlösser anfertigen, die nur von einem (nämlich ihrem) Schlüssel geöffnet werden können. Dann wird sie ihre (offenen!) Schlösser an ihre Freunde verteilen, also auch an Bob.

Bob hat nun ein offenes Schloss von Alice[2], welches er zwar schließen kann, aber ohne passenden Schlüssel nicht wieder zu öffnen vermag (Alice darf ihren Schlüssel natürlich auf keinen Fall bekannt geben).

Also fängt Bob an, seinen Brief zu schreiben, steckt ihn in eine Kiste und verschließt diese mit dem Schloss von Alice.

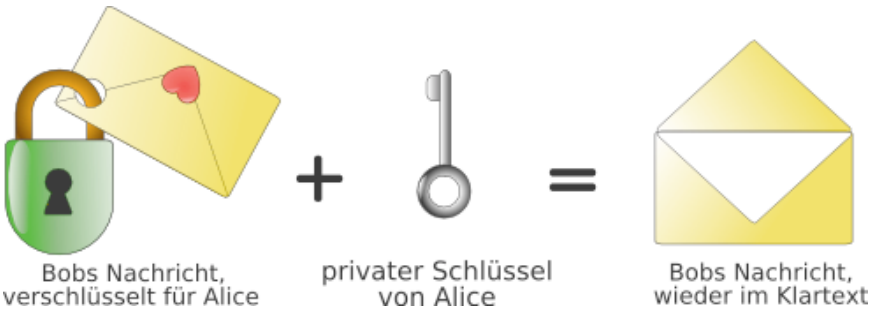
Abgesehen von Alice ist nun niemand mehr in der Lage, die Kiste zu öffnen und den Brief zu lesen. Die Kiste macht sich nun auf die Reise und erreicht irgendwann Alice, welche mit ihrem Schlüssel das Schloss öffnet, den Brief der Kiste entnimmt, liest und froh ist, dass ihr Vater Bobs Brief nicht lesen konnte.

---

[1] Symmetrische Verschlüsselung ist die gebräuchlichste Methode zur Geheimhaltung vor Dritten: Alle Teilnehmenden kennen den geheimen Code, mit dem sowohl ver- als auch entschlüsselt wird. Jeder Mensch, der den gemeinsamen Code kennt, kann die verschlüsselten Daten lesen und verstehen.

[2] Besonders wichtig ist die korrekte Übergabe des Schlosses! Bob muss sich sicher sein können, ein Schloss von Alice zu benutzen. Bestenfalls telefoniert er mit Alice und lässt sich das Schloss genauestens beschreiben.

Alice kann sich absolut sicher sein, dass niemand nach Verschließen der Kiste den Brief lesen konnte. Selbst Bob hatte nicht mehr die Möglichkeit, den Brief zu lesen, geschweige denn zu ändern, da nur Alice den passenden Schlüssel zum Schloss besitzt.



Der angesprochene Vorteil der öffentlichen Schlüsselübertragung besteht also darin, dass prinzipiell jeder ein Schloss von Alice benutzen kann um Kisten zu verschließen, aber nur sie in der Lage ist, diese wieder zu öffnen.

Alice braucht sich zum Gedankenaustausch also nicht unter vier Augen mit Bob zu treffen, was ihr Vater vielleicht gar nicht zulassen würde.
Ein Nachteil besteht allerdings darin, dass Alice sich nicht sicher sein kann ob die Nachricht wirklich von Bob stammt oder ob irgendjemand[3] einfach eines ihrer Schlösser genommen und damit irgendeine Kiste verschlossen hat. Dazu muss sich Bob noch etwas einfallen lassen (z. B. Unterschrift, Blutspritzer, ... → Alice kann damit eine Authentizitätsprüfung[4] vornehmen).

Das Beispiel mit den Schlössern hilft zwar prima bei Verständnis der asymmetrischen Verschlüsselung, ist in der Praxis nur leider nicht wirklich sicher. So würde ein großer Bolzenschneider das Problem des Schlossöffnens für den Vater von Alice vermutlich schon lösen.

Wer bis hierher alles verstanden hat und nun mehr wissen möchte, sollte auf den nächsten Seiten weiter lesen. Dort wird das sehr sichere und in der Praxis häufig genutzte asymmetrisches Verschlüsselungsverfahren **RSA** an einem Beispiel erklärt. Was an mathematischen Verfahren notwendig ist, wird im Abschnitt 2 so weit wie nötig erklärt. Bei schwierigen Operationen helfen der Windows-Taschenrechner und eine Exceltabelle weiter.

Quelle:
http://www.cryptocd.org/online_version/aktuell/doku/windows/asymmetrie/asymmetrie.html
am 5.1.2007

[3] Alice' Vater könnte sein Schloss Bob unterjubeln (als angebliches Schloss von Alice). Bob würde also die Kiste nicht mit dem Schloss von Alice verschliessen, sondern mit dem ihres Vaters. Dieser könnte die Kiste dann bequem mit seinem zugehörigen Schlüssel öffnen, den Brief lesen/manipulieren/zensieren und die Kiste mit Alice' Schloss (welches er ja auch hat, weil es per Definition öffentlich ist) an Alice weiterleiten. Davon würde Alice nichts mitbekommen, da sie wie immer eine Kiste mit einem Brief erhält, verschlossen mit ihrem Schloss. Dieses Vorgehen wird übrigens Man-in-the-Middle-Attacke genannt.

[4] Das Problem der fälschungssicheren Unterschrift ist jedoch auch mit den Werkzeugen der asymmetrischen Verschlüsselung lösbar. Dazu erstellt Alice mit ihrem privaten Schlüssel eine Signatur, anhand derer Bob mit Alice' öffentlichem Schlüssel die Herkunft der Nachricht überprüfen kann.

2. Das RSA-Verfahren

RSA hat den Namen nach den Anfangsbuchstaben der Nachnamen seiner Erfindern RONALD L. **R**IVEST, ADI **S**HAMIR und LEONARD **A**DLEMAN bekommen. **R**IVEST, **S**HAMIR und **A**DLEMAN haben das Verfahren 1977 entwickelt. Es gilt bis heute als sicher.

Die RSA-Verschlüsselung nutzt so genannte **Einweg-Funktionen.** Man kann sich diese Funktionen als mathematische Einbahnstraßen vorstellen. In die eine Richtung (Verschlüsseln) ist die Berechnung ganz einfach. Versucht man den Rechenweg jedoch rückwärts zu beschreiten (Entschlüsseln ohne Schlüssel), wird die Sache sehr viel schwieriger.
Ein Beispiel für eine Einweg-Funktion ist die Multiplikation von Primzahlen. Zur Wiederholung: Eine Primzahl ist nur durch sich selbst und 1 teilbar. Es ist sehr einfach, zwei Primzahlen zu multiplizieren. Nehmen wir 3.259 und 5.431. Das Produkt liefert jeder Taschenrechner in Sekundenbruchteilen, es lautet 17.699.629. Die mathematische Einbahnstraße wurde hierbei in der einfachen (richtigen) Richtung „befahren“. Wenn die Frage nun aber lauten würde: „Gesucht sind alle Teiler der Zahl 17.699.629“ sieht die Sache sehr viel schwieriger aus. Jetzt müssen wir die Einbahnstraße in die andere Richtung zurück fahren. Das ist im Straßenverkehr verboten und in der Mathematik für diese Aufgabe sehr schwierig. 17.699.629 hat nämlich nur vier Teiler: 1 und sich selbst (das gilt immer und für jede Zahl) sowie die beiden Primzahlen, die wir gerade zuvor multipliziert haben. Das bedeutet, wenn man aus dem Produkt 17.699.629 alle Teiler, außer 1 und der Zahl selbst, findet, hat man die Ausgangswerte ermittelt. Das große Problem dabei ist, dass für das Zerlegen einer Zahl in ihre Primfaktoren auf der ganzen Welt keine wirklich guter (d. h. schneller) Algorithmus bekannt ist. Versuchen Sie doch einmal die Faktoren für das Produkt 55.141 zu finden. Kleine Hilfe: es sind natürlich wieder zwei Primzahlen ….

… Na, hatten Sie Erfolg? 55.141 ist keine große Zahl, trotzdem werden Sie die Lösung erst nach einiger Zeit und durch Probieren gefunden haben.. Wenn nicht, die Lösung steht am Ende dieses Textes. Man kann natürlich einwerfen, dass mit Hilfe eines Computers diese Faktoren schnell ermittelt werden können. Richtig ist das aber nur, wenn das Produkt nicht zu groß wird.
Dazu ein Beispiel: Ein 39stelliges Produkt aus zwei 20stelligen Primzahlen wird auf einem aktuellen Computer mit Hilfe einer leistungsfähige (und übrigens teuren) Mathematik-Software in weniger als einer Sekunde in die beiden Primzahl-Faktoren zerlegt. Für eine 41stellige Zahl dauert das Ganze schon etwas mehr als 8 Minuten. Für ein 43stelliges Produkt aus zwei 22stelligen Primzahlen benötigte der Computer fast 19 Minuten. Bei einem 44stelliges Produkt hat das Programm die Berechnung ohne Ergebnis abgebrochen.
In der Praxis wird mit 150stelligen Primfaktoren gearbeitet. Es wird schnell klar, dass auch der schnellste Computer der Welt aus dem 300stelligen Produkt die beiden Primzahl-Faktoren nicht in akzeptabeler Zeit ermitteln und damit die Verschlüsselung ohne Schlüssel knacken kann. Und sollte doch irgendwann einmal ein Computer oder ein Verbund von vielen Computern schnell genug für die Ermittlung der Primfaktoren sein, muss man nur die Anzahl der Stellen der verwendeten Primzahlen erhöhen und schon ist man wieder auf der sicheren Seite.

Dem Besitzer des privaten Schlüssels muss es allerdings möglich sein, die mit dem öffentlichen Schlüssel erzeugte Nachricht relativ leicht zu entschlüsseln. Man bezeichnet Funktionen, die mit einer Zusatzinformation auch rückwärts leicht zu berechnen sind, als **Falltürfunktionen**. Einen solchen mathematischen Zusammenhang gefunden zu haben, ist das Verdienst von RONALD L. **R**IVEST, ADI **S**HAMIR und LEONARD **A**DLEMAN.

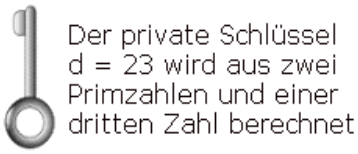
Im Folgenden wird Schritt für Schritt dieses mathematische Verfahren mit ganz kleinen Primzahlen und einer sehr kurzen Nachricht, die nur aus einem Zeichen besteht, erklärt. Keine Angst, wir orientieren uns am Beispiel des Abschnitts 1.

Voraussetzung für das Verständnis ist jedoch die mathematische Rechenoperation ***modulo***. Die Operation *modulo* liefert den **Rest**, der bei einer ganzzahligen Division entsteht.

Beispiele: 20 : 6 = 3 **Rest 2**, denn 3 · 6 =18 ⇨ 18 + **2** = 20 ↳ 20 *mod* 6 = **2**
43 : 5 = 8 **Rest 3**, denn 8 · 5 =40 ⇨ 40 + **3** = 43 ↳ 43 *mod* 5 = **3**
48 : 5 = 9 **Rest 3**, denn 9 · 5 =45 ⇨ 45 + **3** = 48 ↳ 48 *mod* 5 = **3**

Verstanden? Dann kann es jetzt losgehen!

(1) Alice erzeugt zuerst Ihren privaten Schlüssel



Dazu wählt sie zwei Primzahlen, z. B. ***p = 17*** und ***q = 11***.
Diese beiden Zahlen muss Alice geheim halten!
Nun wählt Alice noch eine weitere **Zahl *e***. Nehmen wir an sie wählt ***e = 7***. Dabei sollten ***e*** und ***((p-1) · (q-1))*** teilerfremd sein, d. h. der größte gemeinsamer Teiler von ***e*** und ***((p-1) · (q-1))*** ist 1. Die Wahl einer weiteren Primzahl erhöht die Chance, eine passende Zahl für ***e*** zu finden.
Den eigentlichen **privaten Schlüssel *d*** berechnet Sie dann aus ***p, q*** und ***e*** mit der Formel

$$1 = (e \cdot d) \; mod \; (p - 1) \cdot (q - 1)$$

Mit den eingesetzten Werten für e, p und q lautet die Gleichung:

$$1 = (7 \cdot d) \; mod \; ((17 - 1) \cdot (11 - 1))$$
$$1 = (7 \cdot d) \; mod \; ((16) \cdot (10))$$
$$1 = (7 \cdot d) \; mod \; 160$$

Für ***d* = 23** stimmt die Gleichung: $1 = (7 \cdot 23) \; mod \; 160 = 161 \; mod \; 160$

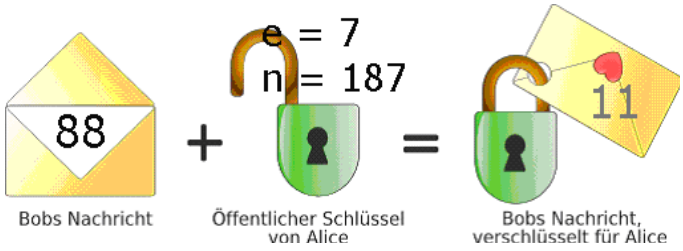
(2) Alice erzeugt dann ihren öffentlichen Schlüssel.
Der öffentliche Schlüssel besteht aus zwei Zahlen. Als erste Zahl verwendet Alice die Zahl ***e = 7*** aus dem ersten Schritt.
Zur Berechnung der zweiten Zahl wählt sie zwei Primzahlen, z. B. *p = 17* und *q = 11*.
Alice multipliziert 17 und 11 und erhält das Produkt ***N = 187***.
Die Zahlen *N* und *e* sind Alice' öffentlicher Schlüssel.

(3) Die zu verschlüsselnde Nachricht wird in eine *Zahl M* umgewandelt. Das kann mit dem ASCII-Code geschehen. Nehmen wir an, Bob möchte Alice den Buchstaben **X** als symbolischen Kuss schicken. Das X hat im ASCII-Code den Dezimalwert 88. Daraus folgt ***M = 88***.

(4) Jetzt kann Bob die *Zahl M* verschlüsseln und erzeugt die *verschlüsselte Nachricht C*.
Die Verschlüsselung von *M* zu *C* erfolgt mit der Formel

$$C = M^{e} \; mod \; N$$

Die öffentlichen Schlüssel *e = 7* und *N = 187* von Alice kennt Bob. Also kann er sie in die Formel einsetzen. Die umgewandelte Zahl *M* ist ebenfalls verfügbar, da es sich um die in eine Zahl umgewandelte Nachricht von Bob handelt.



Bob rechnet also: $C = 88^{7} \; mod \; 187$
Mit der wissenschaftlichen Ansicht des Windows-Rechners erhält man

$$C = 40867559636992 \; mod \; 187 = \mathbf{11}$$

Bob kann diese Nachricht selbst nicht entschlüsseln. Angenommen, er wollte noch einmal überprüfen, ob er wirklich ein X als Symbol für den Kuss geschickt hat, so wäre ihm das nicht möglich. Denn für die Gleichung 11 = $\mathbf{x}^7$ *mod* 187 gibt es unendlich viele Lösungen, z. B. x = 88, x = 275. Allgemein gilt für dieses Beispiel: $x_1 = 88$, $x_2 = x_1 + 187$, …., $x_{n+1} = x_n + 187$.
Bob schickt die verschlüsselte Nachricht *C* = 11 nun an Alice.

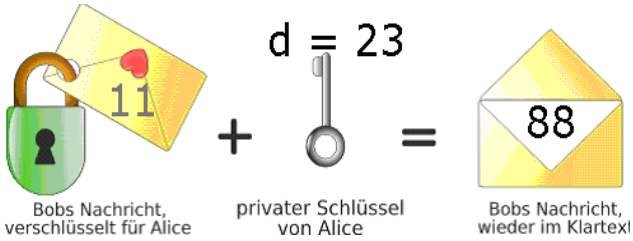
(5) **Alice entschlüsselt die empfangene Nachricht *C***. Dazu benötigt sie ihren privaten Schlüssel d = 23 und den Teil des öffentlichen Schlüssels N = 187. Die Formel zur Berechnung der Originalnachricht lautet:

$$M = C^{d} \; mod \; N$$

Mit eingesetzten Zahlenwerten und dem Windows-Rechner ergibt sich:

$$M = 11^{23} \; mod \; 187$$
$$M = 89543024325523737224653 1 \; mod \; 187$$
$$M = 88$$

Das Zeichen im ASCII-Code mit der Nummer 88 ist das X. Alice hat den symbolischen Kuss erhalten. Hätte ihr Vater den Brief abgefangen, hätte er die Zahl 11 gelesen. Der Zahl 11 ist im ASCII-Code aber gar kein Zeichen zugeordnet. Das ist zwar



Zufall, aber auch bei einem größeren ASCII-Dezimalwert könnte Alice' Vater nur das verschlüsselte Zeichen erkennen. Ein Rückschluss auf die richtige Nachricht ist ohne Kenntnis von Alice' privatem Schlüssel nicht möglich.

Die Wahl der Primzahlen und das Ver- und Entschlüsseln überlassen Alice und Bob ab jetzt einem Computerprogramm. Das Beispiel hat ja gezeigt, dass das Verfahren funktioniert und es wurde erklärt, unter welchen Bedingungen es als sicher angesehen werden kann. Deutlich wurde aber auch, dass RSA sehr rechenintensiv ist. So benötigt RSA 1000mal mehr Zeit als das symmetrische verschlüsselungsverfahren DES. Deshalb ist es für lange Nachrichten nicht geeignet. Symmetrische Verfahren haben aber den großen Nachteil, dass der gemeinsame Schlüssel zwischen Sender und Empfänger ausgetauscht werden muss. Und genau dafür eignet sich RSA hervorragend. Die Schlüssel für die symmetrische Verschlüsselung werden mit dem asymmetrischen Verfahren RSA ausgetauscht. Denn Schlüssel sind im Vergleich zu richtigen Nachrichten sehr kurz.

Literatur: Simon Singh: Geheime Botschaften; Deutscher Taschenbuch Verlag; 6. Auflage; München 2005; S. 436 f
Software: mit ausführlichen Anleitungen und Programmen zum sicheren E-Mail-Verkehr unter www.cryptocd.org und www.gpg4win.de/

Das Produkt 55.141 wurde aus den Primzahlen 823 mal 67 berechnet.

3. Arbeitsblatt zum Testen der RSA-Verschlüsselung

Jetzt dürfen Sie! Legen Sie Ihr Schlüsselpaar fest und übertragen Sie einen selbst gewählten Buchstaben an Ihren Kommunikationspartner!

Für die Wahl des Schlüsselpaares benötigen Sie Primzahlen. Damit die Berechnungen mit dem Windows-Taschenrechner noch möglich sind, verwenden wir nur Primzahlen kleiner als 100. Hier sind alle Primzahlen von 1 – 100:

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

Damit Sie Ihren zu übertragenden Buchstaben in eine Zahl und die Zahl beim Empfang einer Nachricht wieder in einen Buchstaben umwandeln können, ist hier die Liste der deutschen Großbuchstaben des ASCII-Code:

65 = A	**66 = B**	**67 = C**	**68 = D**	**69 = E**	**70 = F**	**71 = G**	**72 = H**	**73 = I**
74 = J	**75 = K**	**76 = L**	**77 = M**	**78 = N**	**79 = O**	**80 = P**	**81 = Q**	**82 = R**
83 = S	**84 = T**	**85 = U**	**86 = V**	**87 = W**	**88 = X**	**89 = Y**	**90 = Z**	

Bei der Berechnung von *d* hilft die Excel-Tabelle „RSA priv Schlüssel d bestimmen.xls“. Sie liefert nach Eingabe von *p*, *q* und *e* den Wert für *d*. Wer das Add-Inn „Analysefunktionen“ installiert hat, kann auch prüfen, ob *e* gut gewählt und zu $((p - 1) \cdot (q - 1)$ teilerfremd ist. Sollte nach der Wahl von *p*, *q* und *e* bei priv. Schlüssel *d* die Ausgabe „**#NV**“ erscheinen, muss man eine andere Zahl für *e* eingeben.

Nachricht mit RSA verschlüsseln	Nachricht mit RSA entschlüsseln
E: Vorbereitungen: ***p*** wählen: _____ (Primzahl-Liste) ***q*** wählen: _____ (Primzahl-Liste) ***e*** wählen: _____ (Primzahl-Liste)	**E: Verschlüsselte Nachricht *C* und privaten Schlüssel *d* bereit halten:** ***C***: _____ ***d***: _____
E: Privaten Schlüssel aus *p*, *q* und *e* ermitteln: ***d*** berechnen: _____ (Excel-Tabelle)	E: Teil *N* des öffentlichen Schlüssels bereit halten ***N***: _____
E: Öffentlichen Schlüssel aus *p*, *q* und *e* ermitteln: ***N*** berechnen: _____ ($N = p \cdot q$) ***e*** notieren: _____ (siehe oben)	**E: *C* entschlüsseln, um *M* zu erhalten:** $M = C^d \bmod N$ ***M*** = _____ (Windows-Taschenrechner)
E: Öffentlichen Schlüssel (*N*, *e*) an Sender übermitteln	**E: Zahl *M* in ASCII-Zeichen umwandeln:** *ASCII-**Zeichen***: _____ (ASCII-Liste)
S: Zeichen, das übertragen werden soll, bestimmen: ASCII-**Zeichen**: _____ ASCII-**Nummer *M***: _____ (ASCII-Liste)	**E: hat die Nachricht entschlüsselt. Das ASCII-Zeichen ist die ursprüngliche Nachricht des Senders.**
S: Zahl *M* verschlüsseln: $C = M^e \bmod N$ = _____ (Win-Rechner)	
S: Verschlüsselte Nachricht *C* an Empfänger E übermitteln	

E steht für die Tätigkeiten, die der **Empfänger** der Nachricht ausführen muss, **S** für die Aufgaben, die der **Sender** der Nachricht abarbeiten muss.