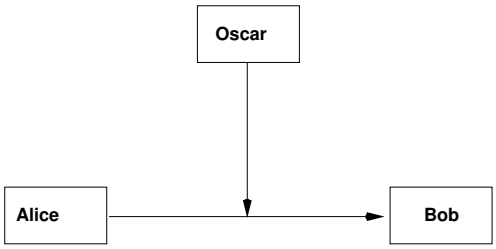
2.5. RSA public key

Es geht hier um das Austauschen von Nachrichten in kodierter Form. Ein naiver Ansatz wäre der Einsatz von Verschlüsselungsbüchern. Da kann der Sender (=Alice) schauen wie man verschlüsselt, der Empfänger (=Bob) sieht nach wie er entschlüsselt. Wenn ein böser Mensch (=Oscar) die Nachricht abfängt, dann kann er bei einer guten Verschlüsselung nichts mit dem abgefangenem Text anfangen.



Das Problem bei diesen symmetrischen (beide haben das gleiche Buch) Verfahren ist, dass irgendwie beide das geheime Codebuch haben müssen. Die Idee beim public key ist, dass jeder die Information zum Verschlüsseln haben kann, da das Entschlüsseln nur mit einer geheimen Information beim Empfänger funktioniert. Dieses Vorgehen wurde 1976 von Diffie und Hellman populär gemacht. Die nun vorgestellte Methode heisst RSA nach den Erfindern von 1978: Rivest, Shamir und Adleman. Sie durften es puplizieren, eine Gruppe in England, die früher dran war, durfte ihr Verfahren aus Sicherheitsgründen nicht publizieren. Für ein public key Verfahren müssen wir 3 Teile betrachten:

- Der publizierte Schlüssel, das ist das öffentliche Codebuch.
- Die Methode für das Verschlüsseln.
- Die Methode für das Entschlüsseln.

2.5.1. Der öffentliche Schlüssel. Der erste Schritt ist das Finden von 2 grossen Primzahlen p, q (100 Dezimalstellen). Das geht relativ einfach indem man zufällig eine 100-stellige Zahl generiert und auf die Primzahleigenschaft testet. Die Zahl $n = pq$ wird publiziert. Nun weiss man (weil man die Primzahlzerlegung kennt)

$$\phi(n) = (p-1)(q-1).$$

Man braucht nun noch eine Zahl a, die teilerfremd zu $\phi(n)$ ist. Auch das geht einfach, man nimmt eine zufällige Zahl (groß) und bestimmt mit dem Euklidschen Algorithmus $ggT(a, \phi(n))$. So findet man schnell (wenige Versuche nötig) eine passsende Zahl und bekommt die Linearkombination

$$1 = ax + \phi(n)y.$$

Auch diese Zahl a wird publiziert.

2.5.2. Verschlüsseln. Die (nach Kodieren) digitale Nachricht wird in Blöcke zerlegt mit weniger Stellen als p oder q. Diese Blöcke werden einzeln verschlüsselt. Ein Block b wird durch

$$b^a \mod n$$

verschlüsselt. Dabei wird der Standardrepräsentant der Kongruenzklasse mod n genommen.

2.5.3. Entschlüsseln. Zum Entschlüsseln genügt es für den empfangenen Block m den Wert

$$m^x \qquad \text{mod } n$$

zu bestimmen. Nun hat der Empfänger die geheime Information $\phi(x)$ und die Linearkombination des ggT, also das x. Damit kann m^x schnell bestimmt werden, und es gilt

$$m^x \equiv (b^a)^x = b^{ax} = b^{1-\phi(n)y} = b(b^{\phi(n)})^{-y} \equiv b \times 1^{-y} = b \qquad \text{mod } n$$

Die letzte Kongruenz gilt nach dem Satz von Euler (2.4.22). Der Satz von Euler verlangt, dass b und n teilerfremd sind, was dadurch erreicht wird, dass a (b ist kleiner als a) kleiner als beide Primzahlen p und q war.

2.5.4. Sicherheit von RSA. Der Algorithmus wird als sicher betrachtet, denn um das geheime x zu bestimmen, muss man den erweiterten ggT Algorithmus mit a und $\phi(n)$ durchführen. Um $\phi(n)$ zu bestimmen, muss man die Faktorisierung von n wissen (siehe 2.4.16). Dies ist für Zahlen mit 200 Dezimalstellen zur Zeit nicht möglich. Es ist wichtig zu beachten, es ist wesentlich leichter eine Zahl auf Primzahleigenschaft zu testen, als sie in Primfaktoren zu zerlegen.

EXAMPLE 2.5.1. RSA

Ein kleines Beispiel mit den Primzahlen $p = 3$ und $q = 41$. Damit ist $n = 123$ und $\phi(n) = 2 \times 40 = 80$. Als zu 80 teilerfremde Zahl wählen wir: $a = 27$. Die Linerakombination des $ggT(a, \phi(n))$ ist

$$1 = 3 \times 27 - 1 \times 80.$$

Unser geheimes x ist dann 3. Wir veröffentlichen: $a = 27, n = 123$.

Um einen Block b zu versenden, berechnet der Sender $b^{27} \mod 123$. Der Empfänger berechnet $m^3 \mod 123$. Als Beispiel versenden wir den Block $b = 05$. Der Sender rechnet

$$5^{27} = 125^9 \equiv 2^9 = 4 \times 128 \equiv 4 \times 5 = 20 \mod 123.$$

Der Empfänger erhält $m = 20$ und rechnet

$$20^3 = 20 \times 400 \equiv 20 \times 31 = 620 \equiv 5 \mod 123.$$