Asymmetrische Verschlüsselung am Beispiel RSA

Wie funktioniert eigentlich die asymmetrische Verschlüsselung?

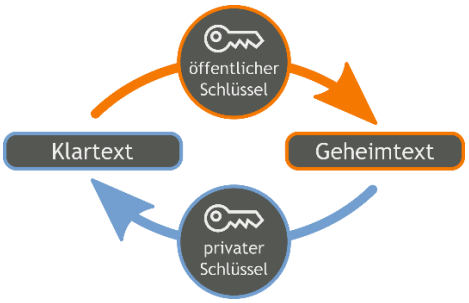
Bis in die 1970er Jahre gab es nur symmetrische Krypto Systeme, bei denen Sender und Empfänger denselben Schlüssel besitzen müssen. Dabei stellt sich das Problem des Schlüsselaustauschs und der Schlüsselverwaltung.

Das erste Public-Key-Verschlüsselungsverfahren war das von Ralph Merkle und Martin Hellman entwickelte Merkle-Hellman-Krypto System. Das MH-Verfahren wurde 1983 von Adi Shamir gebrochen.

Das heute noch verwendete RSA-Verfahren wurde 1977 von Ronald L. Rivest, Adi Shamir und Leonard Adleman am MIT entwickelt.

(14.06.2016 https://de.wikipedia.org/wiki/Public-Key-Verschlüsselungsverfahren)

Bei einem Public-Key-Verschlüsselungsverfahren wird mit einem öffentlichen Schlüssel einen Klartext in einen Geheimtext umgewandelt.



Und nur mit dem geheimen Schlüssel kann dann wieder der Klartext gewonnen werden.

Der geheime Schlüssel muss geheim gehalten werden, und es muss praktisch unmöglich sein, ihn aus dem öffentlichen Schlüssel zu berechnen. Der öffentliche Schlüssel muss jedem zugänglich sein, der eine verschlüsselte Nachricht an den Besitzer des geheimen Schlüssels senden will. Dabei muss sichergestellt sein, dass der öffentliche Schlüssel auch wirklich dem Empfänger zugeordnet ist.

Die Idee des Verfahrens:

Es sei:

- e der Public Key
- d der Private Key
- M ein Zeichnen der zu verschlüsselnden Nachricht.
- C ist das verschlüsselte Zeichen der Nachricht
- N ist die Basis für die Moduloberechungen

Idee: Finde e, N und d so dass das Folgende gilt:

$$C = M^{e} \bmod(N) \quad \text{und} \quad M = C^{d} \bmod(N)$$

Berechnung von Public- und Private Key:

Es sei:

- e der Public Key
- d der Private Key
- M ein Zeichnen der zu verschlüsselnden Nachricht.
- C ist das verschlüsselt Zeichen der Nachricht
- N ist die Basis für die Moduloberechungen

- p und q sind Primzahlen

Public Key berechnen:

1. Zwei Primzahlen wählen:

   p = _____

   q =_____

2. N = p*q = _____

3. Eine weitere Zahl e wählen.
   Voraussetzung:
   e ist *teilerfremd* zu (p-1)*(q-1), also (p-1)*(q-1) darf nicht durch e teilbar sein.
   e ist *teilerfremd* zu N, also e darf weder p noch durch q teilbar sein.

   e = _____

4. Der Public Key besteht aus e und aus N.

Private Key berechnen:

5. Wir müssen nun den Private-Key **d** bestimmt.
   Dazu können wir diese Formel verwenden:

   1 = e*d mod ((p-1)*(q-1))

   Wir können diese Formel nicht nach d auflösen, kennen jedoch e, p und q.
   Nun probieren wir mittels Brute-Force alle Zahlen beginnend von 1, bis wir ein d finden für das die Formel gilt.

   e = _____

Nachricht verschlüsseln:

$C=M^{\wedge e} \bmod(N)$

Nachricht entschlüsseln:

$M=C^{\wedge d} \bmod(N)$

Wieso ist RAS so schwierig zu knacken?

Für die Berechnung von N und d haben wir zwei Primzahlen verwendet: p und q.

$N = p*q$

$1 = e*d \bmod ((p-1)*(q-1))$

e und N werden unsicher übermittelt und sind allen bekannt.

Will man den Schlüssel knacken, so muss man N faktorisieren um p und q herauszufinden.

Wenn N sehr gross ist ~2^2048, dann ist der Aufwand riesig.

Wenn N ~2^(2048 * 8), dann quadriert sich der Aufwand (in etwa).

p;q sollen etwa im Bereich von 2^2048 liegen.

e etwa im Bereich von 2^16+1 (vierte Fermat-Primzahl).

Es gibt zBsp bei Java Funktionen, die solche Primzahlen als Zufallszahl mit einer ungefähren Wahrscheinlichkeit liefern, dass es wirklich eine Primzahl ist.

Sind p oder q keine Primzahlen, dann funktioniert Entschlüsselung nicht.

Sind p oder q sehr klein, dann wird die Faktorisierung einfacher, damit der Code knackbar.