Das RSA-Verfahren

Armin Litzel

Proseminar Kryptographische Protokolle SS 2009

1 Einleitung

RSA steht für die drei Namen Ronald L. **R**ivest, Adi **S**hamir und Leonard **A**dleman und bezeichnet ein von diesen Personen im Jahr 1977 erfundenes asymmetrisches Verschlüselungsverfahren.

Asymmetrische Verschlüsselung wird auch als Public-Key-Kryptographie bezeichnet. Im Gegensatz zur symmetrischen Verschlüsselung kann der Schlüssel zur Verschlüsselung von Klartexten öffentlich gemacht werden ohne die Sicherheit der Kommunikation zu gefährden. Zur Entschlüsselung ist ein entsprechender privater Schlüssel notwendig, der sich aus dem öffentlichen Schlüssel berechnen lässt (umgekehrt ist dies jedoch nicht mit vertretbarem Aufwand möglich). Der private Schlüssel ist deshalb nur dem Schlüsselerzeuger bekannt.

Dadurch entstehen neue Möglichkeiten in der Kommunikation über unsichere Übertragungsmedien (Internet), da hier die Sicherheit durch Einsatz von Public-Key-Verfahren gewährleistet werden kann. Heute werden asymmetrische Verschlüsselungsverfahren häufig verwendet, um Schlüssel für symmetrische Verschlüsselungsverfahren auszutauschen, da diese i.d.R. effizienter sind.

2 Verschlüsselung und Signatur mit RSA

Das RSA-Verfahren kann in Schritten erläutert werden, die zwei Kommunikationspartner durchführen müssen, um eine Information mit Verschlüsselung austauschen zu können. Es soll also eine Nachricht m von A nach B geschickt werden und um dies zu veranschaulichen verwenden wir als Namen der Kommunikationspartner Alice und Bob.

1. Bob wählt zwei Primzahlen p und$q$ so, dass ihr Produkt $n > m$ ist.
2. Bob wählt eine Primzahl e zwischen 1 und $\varphi(n) = (p-1) \cdot (q-1)$ und bestimmt dazu passend eine Zahl $d \in \mathbb{N}$ zwischen 1 und $\varphi(n)$, die dem multiplikativen Inversen modulo n entspricht. Es gilt also $(e \cdot d) \mod \varphi(n) = 1$.
3. Bob sendet seinen öffentlichen Schlüssel (n, e) an Alice .

4. Alice berechnet damit den Geheimtext $c = E(m) = m^e \mod n$ und schickt ihn an Bob.
5. Bob kann die Nachricht mit seinem privaten Schlüssel (n, d) berechnen: $D(c) = c^d \mod n = m$.

Mit dem RSA-Verfahren kann auch der Urheber eines Dokumentes verifiziert werden. Diesen Prozess nennt man digitale Signatur.

Wenn wir davon ausgehen, dass Alice ein Dokument m signieren und an Bob weiterschicken will, dann geschieht dies folgendermaßen:

1. Alice berechnet mit ihrem privaten Schlüssel $s = m^d \mod n$ und schickt s an Bob.
2. Wenn Bob $m = s^e \mod n$ berechnen kann, dann weiß er, dass Alice dieses Signatur erstellt hat.

Um mit RSA Texte und Dokumente zu verschlüsseln, bietet sich der Einsatz von Zeichensätzen an, die 7-8bit umfassen (z.B. ASCII , UTF8). Mit solchen Zeichensätzen, können Texte problemlos in Ziffernblöcke umgewandelt werden.

Die Blocklänge kann sich an der Rechnerarchtiktur orientieren. Da ein 32-Bit Rechner in seinem Register 4 Zeichen speichern kann, bietet es sich hier an, die Blocklänge 4 zu wählen.

Im Folgenden soll die Verschlüsselung der Zeichenfolge »Public-Key erläutert werden:

Wir wählen $p = 251$, $q = 209$ und erhalten $n = p \cdot q = 67519$ sowie $\varphi(n) = 67000$. Desweiteren wählen wir $e = 50253$ und $d = 27917$.

Die Zeichenfolge soll in ASCII-Blöcke (7-bit) unterteilt werden. Da der Dezimalwert eines ASCII-Blocks kleiner sein muss als der Modul n, erhalten wir mit $\log_{128} 67519 \approxeq 2,29$ eine maximale Blockgröße von 2 Zeichen. Die Verschlüsselung illustriert folgende Tabelle:

ASCII	Dezimalwert	Geheimtext C
Pu	$80 \cdot 128^1 + 117 \cdot 128^0 = 10357$	$10357^{50253} \mod 67519 = 17922$
bl	$98 \cdot 128^1 + 108 \cdot 128^0 = 12652$	$12652^{50253} \mod 67519 = 15793$
ic	$105 \cdot 128^1 + 143 \cdot 128^0 = 13538$	$13538^{50253} \mod 67519 = 56395$
-K	$45 \cdot 128^1 + 75 \cdot 128^0 = 5835$	$5835^{50253} \mod 67519 = 10922$
ey	$102 \cdot 128^1 + 121 \cdot 128^0 = 13177$	$13177^{50253} \mod 67519 = 54812$

Die Entschlüsselung ergibt sich dann wie folgt:

C	Dezimalwert	ASCII
17922	$17922^{27917} \mod 67519 = 10357$	$10357 div 128^1 = 80$=> P
		$10357 \mod 128^1 = 117$=> u
15793	$15793^{27917} \mod 67519 = 12652$	$12652 div 128 = 98$ => b
		$12652 \mod 128 = 108$ => l
56395	...	...
10922	...	...
54812	...	...

3 Sicherheit

Die Sicherheit des RSA-Verfahren beruht auf dem Faktorisierungsproblem von ganzen Zahlen. Es ist kein mit bisher technischen Mitteln zu realisierender Algorithmus bekannt, der effizient die Primfaktoren von sehr großen ganzen Zahlen bestimmen kann. Aus diesem Grund ist es bei entsprechend großen Zahlen n zumindest nicht mit vertretbarem Aufwand möglich $\phi(n) = \phi(pq)$ zu berechnen und damit den privaten Schlüssel (d, n) aus dem öffentlichen Schlüssel (e, n) zu bestimmen.

Da bereits erfolgreich eine 1039-bit Zahl faktorisiert wurde, sollte n mindestens 2048 bit lang sein. Desweiteren sollten sich die Primzahlen p und q in ihrer Länge unterscheiden

4 Funktionsweise

Die Funktionsweise des RSA-Verfahren kann durch die Anwendung von zahlentheoretischen Sätzen gezeigt werden. Die wichtigste Grundlage hierfür bilden Restklassen.

Da eine Division durch eine Zahl $n \in N$ genau n verschiedene Reste hervorrufen kann, nämlich die Zahlen $[0, ..., n-1]$, existieren n Restklassen. Die Elemente der Restklassen bilden einen kommutativen Ring. Für das RSA-Verfahren ist die Multiplikation im Restklassenring von Bedeutung:

Korollar 1. $a \mod n \cdot b \mod n \equiv (a \cdot b) \mod n$

Korollar 2. $(a^e) \equiv (a \mod n)^e \mod n$ *für* $e \in \mathbb{N}$

Das RSA-Verfahren kann durch folgenden Satz beschrieben werden:

Satz 4.1. *Seien p und q zwei verschiedene Primzahlen und wird (e, d) so gewählt, dass*

- $e \in [0, ..., \phi(pq) - 1]$ *mit* $ggT(e, \phi(pq)) = 1$
- $d \in [0, ..., \phi(pq) - 1]$ *mit* $e \cdot d \equiv 1 \mod \phi(pq)$

dann gilt für $m \in [0, ..., \phi(pq) - 1]$*:* $m^{ed} \equiv m \mod pq$

Der Beweis folgt mit Hilfe von folgenden Sätzen:

Satz 4.2. *Kleiner Satz von Fermat: Für* $m \in \mathbb{N}$ *und eine Primzahl p mit* $ggT(p, m) = 1$ *gilt:* $m^{p-1} \mod p = 1$

Satz 4.3. *Seien p und q zwei verschiedene Primzahlen. Aus* $a \equiv b \mod p$ *und* $a \equiv b \mod q$ *folgt* $a \equiv b \mod pq$.

Beweis.

1. Es gilt $ed \mod \varphi(pq) = 1$. Aus der Definition der Teilbarkeit folgt, dass es eine ganze Zahl l gibt, mit $0 \leq l < \varphi(pq)$, sodass $ed = l \cdot \varphi(pq) + 1$.
2. Daraus folgt: $m^{ed} \mod pq = m^{l \cdot \varphi(n)+1} \mod pq$.

3. Entweder p ist Teiler von m oder p ist kein Teiler von m.

   a) Wenn p Teiler von m ist, dann gilt offensichtlich $m \mod p = 0$.

$$\begin{aligned} m^{ed} \mod p &= (m \cdot m^{l \cdot \varphi(pq)}) \mod p \\ &= (m \mod p \cdot m^{l \cdot \varphi(pq)} \mod p) \mod p \\ &= (0 \cdot m^{l \cdot \varphi(pq)} \mod p) \mod p \\ &= 0 \\ &= m \mod p \end{aligned}$$

   b) Wenn p kein Teiler von m ist, dann gilt:

$$\begin{aligned} m^{ed} \mod p &= (m \mod p \cdot (m^{p-1})^{l(q-1)} \mod p) \mod p \\ &= (m \mod p \cdot (m^{p-1} \mod p)^{l \cdot (q-1)} \mod p) \mod p \\ &= m \mod p \end{aligned}$$

4. Analog kann für q gezeigt werden: $m^{ed} \mod q = m \mod q$. Da $m^{ed} \equiv m \mod p$ und $m^{ed} \equiv m \mod q$ folgt die Behauptung $m^{ed} \equiv m \mod pq$ aus Satz 2.

□

5 Implementierung

Um RSA implementieren zu können, ist es notwendig eine BigInteger-Bibliothek einzusetzen.

Bei der Schlüsselerzeugung sollten Primzahlen per Zufallsgenerator und Primtest ermittelt werden. Die meisten Algorithmen zur Prüfung auf Primzahlen basieren auf dem Satz von Fermat. Ein sehr effizienter Algorithmus ist der Miller-Rabin-Primtest.

Der Verschlüsselungsexponent e kann auch per Zufallsgenerator und Primtest gewählt werden.

Da $ggT(\varphi(n), e) = 1$ ist, gibt es genau ein positives Zahlenpaar (d,l), welches die Gleichung $1 = e \cdot d + l\varphi(n)$ erfüllt. Mit einer Erweiterung des euklidischen Algorithmus lassen sich die Koeffizienten d und l berechnen. Damit können wir den öffentlichen Schlüssel (d, n) aus dem privaten Schlüssel (e, n) bestimmen.

Satz 5.1. *Sei $r_0 = a, r_1 = b$ sowie $x_0 = 1, x_1 = 0, y_0 = 0, y_1 = 1$.*
Mit $x_{k+1} = q_k x_k + x_{k-1}, y_{k+1} = q_k y_k + y_{k-1}$ und $q_k = r_{k-1} div\, r_k$ gilt:
$r_n = (-1)^n x_n a + (-1)^{n+1} y_n b$

Dieser Satz lässt sich gemäß der Teilbarkeitsdefinition beweisen. Ein entsprechender Beweis wird hier aufgrund des Umfangs ausgelassen. Er kann in Quelle[2] (S.15-16) nachgelesen werden.

Im folgenden soll die Schlüsselerzeugung mit dem erweiterten euklidischen Algorithmus gezeigt werden.

Beispiel 5.1. *Wir wählen* $e = 7$ *und* $\varphi(n) = 20$. *In der Tabelle sind die Reste, Quotienten und Koeffizienten dargestellt, die sich bei Anwendung des euklidischen Algorithmus von den beiden Zahlen ergeben:*

k	0	1	2	3	4
r_k	20	7	6	1	0
q_k		2	1	6	0
x_k	1	0	1	1	7
y_k	0	1	2	3	20

Wir sehen insbesondere, dass $r_3 = 1 = ggT(\varphi(n), e)$. *Mit* $n = 3$ *erhalten wir die "Koeffizientengleichung":* $1 = (-1)^3 \cdot x_3 \cdot \varphi(n) + (-1)^4 \cdot y_3 \cdot e = (-1) \cdot 20 + 3 \cdot 7$

Wir erhalten schließlich d=3. Probe: $(21 \mod 20) = 1$.

Literatur

[1] Claudia Eckert: IT-Sicherheit: Konzepte, Verfahren, Protokolle. R. Oldenbourg-Verlag, 2007 (5.überarbeitete Auflage).

[2] Johannes Buchmann: Einführung in die Kryptographie. Springer, Berlin, 2004 (3. erweiterte Auflage).