

Verschlüsselung mit dem RSA-Verfahren

Finn Sohst
 Fachbereich Informatik an der Universität Hamburg
 Brahmsallee 27
 20144 Hamburg
 040/272736

5sohst@informatik.uni-hamburg.de

ABSTRACT

In der heutigen Welt spielt Datensicherheit eine immer grössere Rolle. Symmetrische Verschlüsselungsverfahren sind nicht mehr sicher genug. Aus diesem Grund werden heutzutage asymmetrische Verschlüsselungen benutzt, zu denen das RSA-Verfahren zählt. In diesem Paper wird die mathematische Funktionsweise des RSA-Verfahrens erläutert. Einleitend wird ein kurzer Überblick über die Entstehung des Verfahrens gegeben sowie der Unterschied zwischen symmetrischer und asymmetrischer Verschlüsselung beschrieben.

Allgemeine Begriffe

RSA-Verschlüsselung, asymmetrische Verschlüsselung

Schlüsselworte

Euklidischer Algorithmus, Multiplikatives Inverses, RSA, Ronald L. Rivest, Adi Shamir, Leonard Adleman, Encryption, Primzahlen, Public Key

1. EINLEITUNG

Eine der einfachsten Verschlüsselungen ist das Caesar Chiffre. Hierbei werden die Buchstaben des Alphabets um einen bestimmten Wert verschoben und der Klartext durch dieses verschobene Alphabet ersetzt.

A	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
B	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c

Tabelle 1. A ist das Klartextalphabet, B ist das Caesar Alphabet

So würde beispielsweise das Wort "Hallo" verschlüsselt "Kdoor" lauten.

Bei der symmetrischen Verschlüsselung wird zum Entschlüsseln der gleiche Schlüssel verwendet wie zur Verschlüsselung. Somit muss dieser Schlüssel gleichzeitig beim Absender und beim Empfänger vorhanden sein.

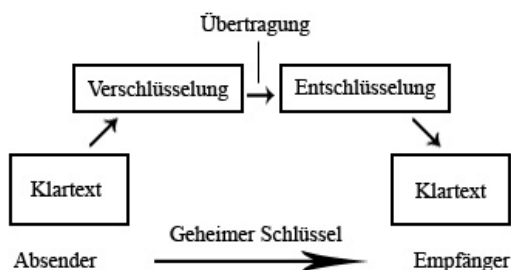
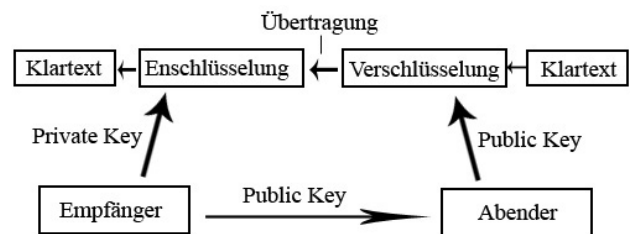


Bild 1. Symmetrische Verschlüsselung

Durch diesen Umstand ist es erforderlich, dass der Schlüssel zwischen Absender und Empfänger ausgetauscht wird. Dieser Austausch stellt das Sicherheitsrisiko beim symmetrischen Verfahren dar, da der Schlüssel zum Zeitpunkt der Übertragung von einer dritten Person gespeichert werden könnte. Aus diesem Grund entwickelten die drei Mathematiker Ronald L. Rivest, Adi Shamir und Leonard Adleman 1977 ein Verfahren, bei dem der geheime Schlüssel nicht mehr ausgetauscht werden musste. Die Anfangsbuchstaben ihrer Nachnamen bilden den Namen des Verfahrens – RSA.



2. DAS RSA-VERFAHREN

Das RSA Verfahren basiert auf dem Problem der Faktorisierung sehr grosser Zahlen. Während es sehr einfach ist, zwei grosse Primzahlen zu multiplizieren, ist es sehr schwierig, aus dem Produkt dieser Zahlen auf sie zu schliessen. RSA wurde erfunden, um das Problem des geheimen Schlüssels zu umgehen, der beim symmetrischen Verfahren zwei Personen zugänglich sein muss.

Bild 2. Asymmetrische Verschlüsselung

In Bild 2 ist zu sehen, dass der Private Key (Private Schlüssel) nur noch einer Person zugänglich ist. Mit dem Public Key (Öffentlicher Schlüssel) kann nur noch verschlüsselt, nicht aber entschlüsselt werden. Aus diesem Grund kann er auch jeder Person zugänglich sein. Es kann allerdings auch mit dem Private Key verschlüsselt und mit dem Public Key entschlüsselt werden. Dies wird normalerweise zur Authentifizierung in Netzwerken benutzt, um die Echtheit einer Nachricht zu versichern. Denn nur wenn der Empfänger die Nachricht mit dem Public Key entschlüsseln kann, ist sicher, dass der Absender mit dem dazugehörigen Private Key verschlüsselt hat.

2.1 Der Algorithmus

Zuerst werden zwei möglichst grosse und etwa gleich lange Primzahlen benötigt um das sog. Systemmodul (N) zu berechnen.

$$N = p * q; p, q \in P$$

dannach wird $\phi(N)$ berechnet, wobei ϕ die Eulersche Phi-Funktion ist. Die Phi Funktion gibt an, wieviele Zahlen

zwischen 1 und n-1 teilerfremd zu n sind. Bei Primzahlen ist sie immer n-1.

$$\phi(N) = (q-1) * (p-1)$$

Nun wird der öffentliche Schlüssel E zufällig gewählt, wobei zu beachten ist, dass E und $\phi(N)$ teilerfremd sind.

Also gilt:

$$\text{ggT}(E, \phi(N)) = 1; 1 < E < \phi(N)$$

Der private Schlüssel D muss das multiplikative Inverse des öffentlichen Schlüssels sein. Damit erhält man folgende Bedingung:

$$(E * D) \bmod \phi(N) \equiv 1$$

Das Multiplikative Inverse lässt sich mit Hilfe des erweiterten Euklidischen Algorithmus berechnen.

Die Verschlüsselung funktioniert folgendermaßen:

$$C_i = M_i^E \bmod N$$

Wobei M_i in Zeichen der zu verschlüsselnden Nachricht ist.

Ähnlich funktioniert die Entschlüsselung:

$$M_i = C_i^D \bmod N$$

2.2 Ein Beispiel

In diesem Beispiel werden zu Anschauungszwecken sehr niedrige Zahlen benutzt.

2.2.1 Schlüsselerstellung

Man wähle 2 Primzahlen als p und q

$$p = 13$$

$$q = 17$$

$$N = p * q = 13 * 17 = 221$$

Nun wendet man die Eulersche Phi-Funktion an

$$\phi(N) = (p-1) * (q-1) = 192$$

Nun wählt man für den öffentlichen Schlüssel E eine Zahl zwischen 1 und $\phi(N)$, die teilerfremd zu $\phi(N)$ ist.

$$\text{ggT}(E, \phi(N)) = 1; 1 < E < \phi(N); \text{Wähle } E = 71$$

D soll das multiplikative Inverse von E sein. Hierzu prüft man zuerst mit dem Erweiterten Euklidischen Algorithmus die Teilerfremdheit von E und $\phi(N)$.

$$192 = 2 * 71 + 50$$

$$71 = 1 * 50 + 21$$

$$50 = 2 * 21 + 8$$

$$21 = 2 * 8 + 5$$

$$8 = 1 * 5 + 3$$

$$5 = 1 * 3 + 2$$

$$3 = 1 * 2 + 1$$

$$2 = 2 * 1 + 0$$

Wie erwartet ist der $\text{ggT}(71, 192) = 1$

Es gilt: wenn $\text{ggT}(a, b) = 1$ dann gibt es (c, d) mit denen gilt

$$a * c - b * d = 1$$

Daraus kann man schliessen, dass es (D, k) geben muss, mit denen gilt

$$k * 192 - D * 71 = 1$$

wobei D das multiplikative Inverse von 71, also der private Schlüssel ist.

Nun berechnet man mit Hilfe der Zahlen aus dem Euklidischen Algorithmus D:

$$1 = 3 - 1 * 2$$

$$= 3 - 1 * (5 - 1 * 3) = -1 * 5 + 2 * 3$$

$$= -1 * 5 + 2 * (8 - 1 * 5) = 2 * 8 - 3 * 5$$

$$= 2 * 8 - 3 * (21 - 2 * 8) = -3 * 21 + 8 * 8$$

$$= -3 * 21 + 8 * (50 - 2 * 21) = 8 * 50 - 19 * 21$$

$$= 8 * 50 - 19 * (71 - 1 * 50) = -19 * 71 + 27 * 50$$

$$= -19 * 71 + 27 * (192 - 2 * 71) = 27 * 192 - 73 * 71$$

$$\Rightarrow k = 27, D' = -73$$

Da D aber nicht negativ sein, darf subtrahiert man D' von $\phi(N)$

$$D = 192 - 73 = 119$$

Die Überprüfung:

$$D = 119 \text{ denn } 119 * 71 = 44 * 192 + 1$$

Nun hat man alle Schlüssel berechnet:

Öffentlicher Schlüssel: E = 71 mit N = 221

Geheimer Schlüssel: D = 119 mit N = 221

2.2.2 Verschlüsselung

Nun verschlüsselt man eine Nachricht mit Hilfe der Formel

$$C_i = M_i^E \bmod N$$

In diesem Beispiel Ordnen wir den Buchstaben des Alphabets einfach ihre Position als Zahl zu:

$$a = 1, b = 2, c = 3, \dots$$

Mit dieser Zuordnung verschlüsseln wir das Wort "hallo"

$$\text{hallo} = 8|1|12|12|15$$

$$C1 = 8^{71} \bmod 221 = 83$$

$$C2 = 1^{71} \bmod 221 = 1$$

$$C3 = 12^{71} \bmod 221 = 194$$

$$C4 = 12^{71} \bmod 221 = 194$$

$$C5 = 15^{71} \bmod 221 = 59$$

Verschlüsselt lautet das Wort "hallo" also

$$83|1|194|194|59$$

2.2.3 Entschlüsselung

Nun entschlüsseln wir mit der Formel $M_i = C_i^D \bmod N$

$$M1 = 83^{119} \bmod 221 = 8$$

$$M2 = 1^{119} \bmod 221 = 1$$

$$M3 = 194^{119} \bmod 221 = 12$$

$$M4 = 194^{119} \bmod 221 = 12$$

$$M5 = 59^{119} \bmod 221 = 15$$

$$\Rightarrow 8|1|12|12|15 = \text{Hallo}$$

3.SICHERHEIT UND ZUKUNFT

3.1 Faktorisierung

Die Firma RSA Security[1] gibt Preisgelder für die Primzahlfaktorisation von RSA Zahlen aus. Das momentan höchste Preisgeld von \$200.000 ist für die Faktorisierung einer 1024-Bit RSA Zahl ausgeschrieben. Diese Zahl hat 617 Stellen und es ist unwahrscheinlich, dass sie mit heutigen Rechnern faktorisiert werden kann. Am 4 November 2005 wurden allerdings \$20.000 an ein deutsches Wissenschaftlerteam gezahlt, nachdem sie eine 640-Bit RSA Zahl mit 193 Stellen faktorisiert hatten.

3.2 Die Zukunft

Momentan ist das RSA-Verfahren sehr sicher, da mit heutiger Technik keine 1024-Bit Zahlen faktorisiert werden können. In Zukunft könnte das Verfahren jedoch unsicher werden. 1994 entwickelte der Wissenschaftler Peter W. Shor einen Algorithmus für Quantencomputer, mit dem es möglich ist, eine Primzahlfaktorisation in kürzester Zeit durchzuführen[2]. Da Quantencomputer heutzutage aber technisch noch nicht realisierbar sind, kann dieser Algorithmus noch nicht zum Einsatz kommen.

4.REFERENZEN

- [1] <http://www.rsasecurity.com/rsalabs/node.asp?id=2093>
- [2] Peter W. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. In *SIAM J.Sci.Statist.Comput.* 26 (1997) 1484 , 1997
- [3] Hagen Ploog und Dirk Timmermann, Mobile Sicherheit durch effiziente Public-Key-Verschlüsselung, Universität Rostock, 2001
- [4] Ryan Knighlinger, RSA Encryption Algorithm
- [5] <http://kightlinger.net/rsa/Ryan%20Kightlinger%20-%20Number%20Theory%20-%20RSA%20Encryption%20Algorithm.pdf>, 2005.
- [6] Mihir Bellare and Phillip Rogaway, Optimal Asymmetric Encryption – How to Encrypt with RSA, In *Advances in Cryptology - Eurocrypt 94 Proceedings, Lecture Notes in Computer Science Vol. 950, A. De Santis ed., Springer-Verlag*, 1994