Einführung zu RSA

Andreas Zweili, Ismail Cadaroski, Ivan Hörler, Michael Stratighiou

9. Januar 2017

Inhaltsverzeichnis

1 Einführung **3**

1.1 Geschichte 3

1.2 Verwendung 4

2 Öffentlicher und Privater Schlüssel **5**

2.1 Schlüsselkontruktion 5

2.2 Konstruktion N 6

2.3 Konstruktion m 6

2.4 Konstruktion e 6

2.5 Konstruktion (d) 7

3 Verschlüsselung **8**

3.1 Der eigentliche Akt der Verschlüsselung 8

3.2 Die Übermittlung 9

4 Entschlüsselung **9**

5 Schwachstellen **10**

5.1 Brut-force 10

5.2 Fakturierung durch die Kenntnis von N 10

5.3 Berechnung von φN ohne Fakturierung von N 11

5.4 zu kleine Multiplikator-Primzahlen 11

5.5 Gleiche φ N 12

5.6 Riehmann hypotese 12

5.7 Social Engineering 12

6 Referenzen **13**

1 Einführung

Diese Arbeit wird eine Einführung zu dem Verschlüsselungsalgorithmus RSA geben. Anhand von vereinfachten Rechnungen wird die Funktion des Algorthmus veranschaulicht und erklärt. In der Realität sind die verwendeten Zahlen jedoch um ein x-faches grösser. Die nachfolgende Zahl ist 1024 Bit gross. Der Leser kann sich also ungefähr vorstellen wie gross die Zahlen sind wenn die heutige empfohlene Grösse bei 4096 Bit liegt.

RSA-1024 Primzahl

```
13506641086599522334960321627880596993888147560566
70275244851438515265106048595338339402871505719094
41798207282164471551373680419703964191743046496589
27425623934102086438320211037295872576235850964311
05640735015081875106765946292055636855294752135008
52879416377328533906109750544334999811150056977236
890927563
```

1.1 Geschichte

Im Jahre 1976 wurde von Whitfield Diffie und Martin Hellman eine Theorie zu Publickey-Kryptographie veröffentlicht [**?**]. In welcher sie ein Konzept Namens FFalltürpräsentieren. Dabei handelt es sich um mathematische Probleme welche in eine Richtung sehr aufwändig und in die andere Richtung viel einfacher zu lösen sind.

Ronald L. Rivest, Adi Shamir und Leonard Adleman wollten nach der Veröffentlichung der Theorie von Herrn Diffie und Herrn Hellman beweisen das solche Falltüren nicht existieren. Dabei entdeckten sie jedoch genau solch eine Falltür daraus entwickelten sie dann den RSA Algorithmus welchen sie 1977 vorstellten [**?**]. RSA steht dabei für die Anfangsbuchstaben ihrer Familiennamen.

Im Jahre 2002 erhielten sie den Turing-Award für ihre Arbeit auf dem Gebiet der Kryptographie. Welcher oft als Nobel Preis für Informatik bezeichnet wird.

1.2 Verwendung

RSA wird heute in eine Vielzahl an Programmen eingesetzt. Von besonderer Wichtigkeit sind hier folgende Systeme zu Erwähnen.

Bankkarten nach dem EMV Standard

Dieser Standard definiert wie der Chip auf den Karten zu funktionieren hat und wie die Authentifizierung gegenüber den Bankautomaten funktioniert.

HTTPS (TLS und X.509-Zertifikate)

HTTPS garantiert das die Zugriffe auf Website welche es unterstützen, vor Manipulationen sowie Spionage von Unbefugten geschützt sind. Dies ist insbesondere bei eBanking oder Websiten mit Logins essentiel wichtig. Ansonsten ist es ein Leichtes Konten zu übernehmen.

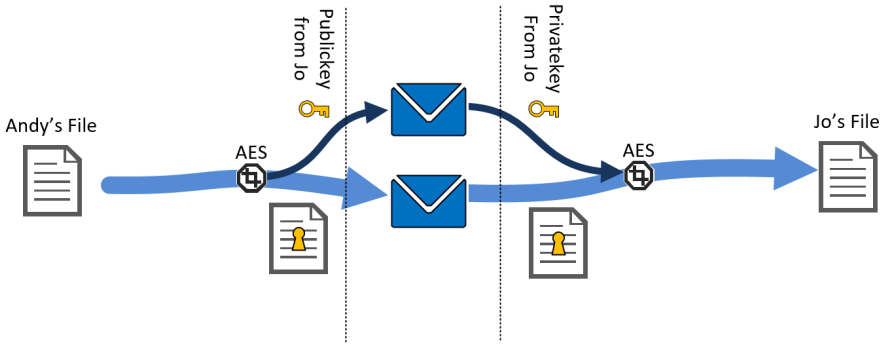
SSH (Secure Shell)

SSH ist ein Protokoll mit welchem man remote auf Unix Systeme Zugreifen kann. Am häufigsten wird es genutzt zur Administrierung von Servern oder zur Übertragung von Dateien.

OpenPGP

OpenPGP ist ein Verschlüsselungsverfahren welches hauptsächlich bei der Verschlüsselung von Emails verwendet wird. Abseits davon wird es auch zur Signierung von Dateien eingesetzt.

Zusätzlich sollte noch erwähnt werden das RSA in den meisten Fällen nicht alleine eingesetzt wird da die Performance von RSA im Vergleich zu symetrischen Verfahren sehr viel schlechter ist. Deshalb wird RSA oftmals nur zum Schlüsseltausch eingesetzt und eine symetrische Verschlüsselung zum Verschlüsseln der eigentlichen Daten.

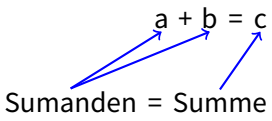


2 Öffentlicher und Privater Schlüssel

Als erster Schritt muss ein öffentlicher und privater Schlüssel (sozusagen ein Schlüsselpaar), konstruiert werden. Dazu wählen wir 2 zwei zufällie Primzahlen die wir in unserem Beispiel zur einfachheit halber klein halten und fangen mit der Konstruktion an.

2.1 Schlüsselkontruktion

Beispiel für Isi:



In den folgenden Seiten berechnen wir :
N = Privatschlüsselanteil
p = primzahl
q = primzahl
e = Teilerfremder Wert
d = modular inverse

In dem e+n den öffentlichen und d+n den privaten Schlüssel bilden.

2.2 Konstruktion *N*

Es werden zwei verschiedene Primzahlen p = 7 und q = 11 gewählt und das Produkt daraus gerechnet, welches wir N nennen.

$$N = p \cdot q$$
$$77 = 7 \cdot 11$$
$$N = 77$$

2.3 Konstruktion *m*

Danach rechnen wir Phi von N um die Anzahl der teilerfremden Zahlen zu berechnen. Da p und q Primzahlen sind wissen wir das Phi von p = p – 1 und Phi von q = q – 1 ist.

$$\varphi(N) = \varphi(p \cdot q)$$
$$\varphi(N) = \varphi(p) \cdot \varphi(q)$$
$$\varphi(N) = (p - 1) \cdot (q - 1)$$
$$\varphi(N) = (7 - 1) \cdot (11 - 1)$$
$$\varphi(N) = 60$$

2.4 Konstruktion *e*

Wir bestimmen eine zu $m = 60$ teilerfremde Primzahl mit $1 < e < m$ mit dem ggT(e,m)=1
Wir nehmen in unserem Beispiel 7 da sie nicht durch 60 teilbar ist und beide den ggT 1 besitzten.

2.5 Konstruktion *(*d)

Zuvor haben wir e mit der Eigenschaft (ggT(e, φ(N) = 1) bestummen, dies bedeutet, wenn wir (modφ(N)) rechnen, hat e einen inversis.
Wir bestimmen d mit e $*$ d $\equiv$ 1modφ(N) und formen dies nach d = e^{-1}modφ(N)
Aus e^{-1}modφ(N) $\equiv$ 7^{-1}modφ(77) berechnen wir die inverse, indem wir erstmal aus 60 und 7 den euklidischen Algorithmus formen, als wollten wir den ggT dieser Zahlen ermitteln. Wir wissen schon dass der ggT(60,7) 1 ist.

$$60 = 8 \cdot 7 + 4$$
$$7 = 1 \cdot 4 + 3$$
$$4 = 1 \cdot 3 + 1$$
$$3 = 3 \cdot 1 + 0$$

Der erweiterte euklidische Algorithmus besteht nun darin, ausgehend von der vorletzten Seite, diese Rechenschritte ”von unten nach oben” in der folgenden Weise aufzurollen, indem die einzelnen Zeilen nach den Resten aufgelöst und diese nacheinander eingesetzt werden:

$$1 = 4 - 1 \cdot 3$$
$$1 = 4 - 1 \cdot (7 - 1 \cdot 4)$$
$$1 = 4 - 1 \cdot 7 + 1 \cdot 4$$
$$1 = (-1) \cdot 7 + 2 \cdot 4$$
$$1 = (-1) \cdot 7 + 2 \cdot (60 - 8 \cdot 7)$$

Zu beachten ist, dass wir alle Klammern, jedoch nicht alle Produkte ausmultiplizieren

Da unsere Inverse positiv und kleiner 60 sein soll, addieren wir 60 $*$ 7 auf der linken Seite. Die rechte Seite verändert sich nicht, da wir mod 60 rechnen.

$$1 = 2 \cdot 60 - 17 \cdot 7 \mid (\text{mod})$$

$$1 = 43 \cdot 7 \bmod 60 \mid (60 - 17 = 43)$$

$$1 = d^e \bmod n / \mid \text{Umformen nach } d$$

$$d = 7^{-1} \bmod 60$$

$$d = 43$$

Danach lösen wir die Gleichung nach d auf.

3 Verschlüsselung

Im Beispiel der Schlüsselkonstruktion werden die Variablen e und N als öffentlicher Schlüssel festgelegt. Dieser wird benötigt um eine Nachricht für den dafür entsprechenden Empfänger zu verschlüsseln. Mit der daraus resultierenden Zahl sowie dem privaten Schlüssel, welcher aus den Variablen d und N besteht, kann die Nachricht wieder entschlüsselt werden.

In unserem Beispiel lautet der private Schlüssel also: 43 + 77 und der öffentliche Schlüssel: 7 + 77

3.1 Der eigentliche Akt der Verschlüsselung

Wollen wir nun eine Nachricht mit dem öffentlichen Schlüssel verschlüsseln, so das sie also nur noch für den Empfänger mit dem entsprechenden privaten Schlüssel zu entschlüsseln ist, gehen wir folgendermassen vor:

Wir kennen die beiden Zahlen des öffentlichen Schlüssels: 7 + 77

Unsere zu verschlüsselnde Nachricht x: 45 (muss kleiner sein als N) (Wie bereits in einem früheren Kapitel erwähnt sind solche öffentliche Schlüssel Primzahlen mit mehreren hundert Stellen, somit ist diese Regel im Normalfall irrelevant. Da wir aber in unserem Beispiel keine so grossen Primzahlen verwenden müssen

wir diesen Punkt beachten um sicherzustellen das wir auch ein korrektes Ergebnis erhalten.)

Die Nachricht wird nun mit folgender Formel verschlüsselt:

$$y = x^a \mod n$$
$$y = 45^{43} \mod 77$$
$$y = 45$$

3.2 Die Übermittlung

TODO: Muss ergänzt werden.

45 (y) ist nun also unsere Verschlüsselte Nachricht, welche nun an den Empfänger übermittelt wird.

4 Entschlüsselung

Um die Nachricht zu entschlüsseln muss zuerst d errechnet werden, dies geschieht mithilfe des erweiterten euklidischen Algorithmus. Diese Berechnung wurde bereits im Kapitel Schlüsselerzeugung erledigt. Unsere gesuchte Zahl lautet demnach 43 (d)

Da nun alle benötigten Variablen bekannt sind kann die Nachricht mit folgender Formel entschlüsselt werden.

$$x = y^d \mod n$$
$$x = 45^{43} \mod 77$$
$$x = 45$$

5 Schwachstellen

Obwohl schon einige verkündet haben die RSA Verschlüsselung geknackt zu haben ist es bisher noch niemandem gelungen einer Überprüfung stand zu halten. Es gibt aber durchaus realistische ideen wie der Code zerbrochen werden kann, nachgehend stellen wir die Wichtigsten Methoden vor.

5.1 Brut-force

Die Methode alle möglichen Primzahlen von $\varphi = (p - 1) \cdot (q - 1)$ auszuprobieren gilt als nicht einfacher als N zu Faktorisieren.

5.2 Fakturierung durch die Kenntnis von *N*

Weil die Faktoren von N den φN ermitteln lassen kann auch d ermittelt werden. Die Erfinder RSA selbst, berechneten anhand eines Algorithmus von Richard Schroeppel und der Annahme das ein Annäherungsschritt 1ms benötigt die Zerlegung von:

Zeichen	**Operationen**	**Zeit**
50	$1.4 \cdot 10^10$	3.9 Stunden
75	$9.0 \cdot 10^12$	104 Tage
100	$2.3 \cdot 10^15$	74 Jahre
200	$1.2 \cdot 10^23$	$3.8 \cdot 10^9$ Jahre

1. Diese Berechnungen der Entschlüsselungs-Zeiten sind überholt. (stand 1978)
2. 1996 schreibt der Prof. Johannes Buchmann von der Universität Saarbrücken das ein Parallelisiertes Netz von 250 Rechnern auf dem Campusareal für eine 130 Stellige Zahl mehrere Wochen benötigt und sich mit mit drei zusätzlichen Dezimalstellen verdoppelt.
3. 2003 veröffentlichte Adi Shamir und Eran Tromer einen technischen Report wie ein RSA Schlüssel von 1024 bit in unter einem Jahr gebrochen werden kann. [**?**]

Diese drei Beispiele zeigen auf wie unvorhersehbar die Standhaftigkeit eines Schlüssels in Bezug auf Zeit ist.

Die Formel zur Zerlegung von φN lautet:

$$\varphi = 2 \cdot \text{kgV}\left(\frac{p-1}{2}, \frac{q-1}{2}\right)$$

5.3 Berechnung von φN ohne Fakturierung von N

Natürlich lässt sich φN auch ohne Fakturierung von N ermitteln wenn d bekannt ist oder ermittelt werden kann. Da d jedoch ein multiplikator von φN ist, ist sein wert nicht leichter zu ermitteln als die Fakturierung von N ist.

5.4 zu kleine Multiplikator-Primzahlen

Da die Sicherheit von RSA darauf beruht dass die Fakturierung von Primzahlen Zeit benötigt, ist sie auch nur so stark wie die Grösse der Primzahl q die Multipliziert mit p den Modulus ergibt. Ist q oder p kleiner als 100 Stellen, wird daraus nicht ein Schlüssel > $10^2$00 entstehen und damit die Verschlüsselung zwar schneller geschehen aber sie ist auch gefährdeter durch Brute-force Attacken oder Fakturierung zerlegt zu werden.

5.5 Gleiche φ *N*

5.6 Riehmann hypotese

Die Riehmann Hypothese beschreibt ein bisher ungelöstes Mathematisches Problem. Sollte sich die Theorie der Reihmann Hypothese bewarheiten könnten daraus Primzahlen abgeleitet werden auf dessen Basis die Zerlegung von *N* einfacher und schneller ausgeführt werden kann.

5.7 Social Engineering

Durch das Abfangen einer Nachricht kann ein Angreifer damit noch nichts anfangen da sie mit dem Schlüssel des Empfängers Verschlüsselt ist. Möchte er diese nun entschlüsseln muss er an den Schlüssel des Empfängers kommen. Dazu kann er die Datei wiederum mit einem Ihm bekannten Schlüssel verschlüsseln und sie dem Empfänger erneut und gegebenenfalls unter Verschleierung seiner Identität zustellen. Dieser wird nun die Datei mit seinem Schlüssel entschlüsseln und nichts damit anfangen können da sie immer noch mit dem Schlüssel des Angreifers verschlüsselt ist. Bringt nun der Angreifer durch Geschick den Empfänger dazu Ihm diese entschlüsselte vermeintlich defekte Datei zuzusenden kann er sie mit seinem Schlüssel entschlüsseln und den Inhalt lesen.

6 Referenzen