Silverberg - Gymnasium Bedburg

RSA Verschlüsselung unter Berücksichtigung des kryptographischen Ursprungs, der Abgrenzung zu anderen Verschlüsselungsverfahren und dem besonderen Schwerpunkt einer praktischen Umsetzung anhand eines eigens geschriebenen Programms

Autor: Andreas Link

Kurs: Mathematik Grundkurs

Kursleiter: Herr Rosenbaum

Abgabetermin: 03.03.2010

Vorwort

Zu Beginn meiner Facharbeit möchte ich insofern einen Überblick geben, als dass ich einen Einstieg in die Thematik ermögliche, mein Vorgehen beschreibe, dabei einen kurzen Bezug auf meine Gliederung herstelle und schließlich das eigentliche Ziel meiner Facharbeit erläutere.
In meiner Arbeit behandle ich die Thematik der „RSA Verschlüsselung unter Berücksichtigung des kryptographischen Ursprungs, der Abgrenzung zu anderen Verschlüsselungsverfahren und dem besonderen Schwerpunkt einer praktischen Umsetzung anhand eines eigens geschriebenen Programms“.
Hierbei möchte ich in Punkt 1 zunächst Bezug auf die Ursprünge der Kryptographie nehmen, wobei die Schwächen der ersten Verschlüsselungsmethoden gezeigt und das daraus resultierende Bedürfnis nach einem weitreichenderen Verfahren erläutert werden soll. Im Anschluss daran werde ich die Unterschiede zweier Verschlüsselungsverfahren, die sich im alltäglichen Gebrauch des 21. Jahrhunderts etabliert haben, unter Punkt 2 erklären.
Das Ziel meiner Facharbeit ist, ein Programm zu entwickeln, dessen Voraussetzung es sein wird, sich immer wieder neu aufkommenden Herausforderungen zu stellen und auf diese Weise herauszufinden, ob es mir möglich sein wird, die Grundlagen des so genannten „RSA Verfahren“ praktisch umzusetzen. Dabei werde ich zunächst kurz, unter Punkt 3 meiner Arbeit, auf die mathematischen Grundlagen für das Verständnis von RSA eingehen, mein Projekt daraufhin in Punkt 4 erläutern und schließlich meine Arbeit unter Punkt 5 abschließen, indem ich auf die Sicherheit und die damit verbundene Zukunftsorientierung des Verfahrens eingehe und abrundend ein Fazit vornehme.

Inhaltsverzeichnis:

1.Einführung

1.1 Warum bestand zu Zeiten Cäsars schon das Interesse Daten zu verschlüsseln und auf welche Weise versuchte man eine Lösung diesbezüglich zu finden?

Schon im Jahre *100 v.Chr.*, zu Zeiten „*Gaius Iulius Cäsars*“[1], entwickelte sich das Bedürfnis politisch einflussreicher Personen, Daten zu verschlüsseln und sie dadurch für Unbefugte unbrauchbar zu machen. Da zu diesem Zeitpunkt noch keinerlei Möglichkeit bestand, Nachrichten in einem möglichst kurzen Zeitraum über große Distanzen zu übermitteln, wie wir es heute mit Hilfe des Internets tagtäglich praktizieren, musste man auf Boten zurückgreifen, die über lange Reisen hinweg Nachrichten persönlich an den zuvor bestimmten Empfänger überbrachten. Diese Methode, Informationsaustausch zu betreiben, zeigte sich als äußerst langwierig und unsicher. Aufgrund der Tatsache, dass sensible Informationen für sämtliche politische Bereiche nur auf diesem Wege übertragen werden konnten, bestand ein großes Interesse der feindlichen Mächte, diese abzufangen und dadurch z.B Aufschluss über das geplante kriegerische Vorgehen des Feindes zu erhalten. Somit boten auch Großmächte, trotz ihrer kriegerischen Überlegenheit, eine ausreichende Angriffsfläche für Feinde, konnten sie keinen gesicherten Datenaustausch zwischen ihnen selbst und Verbündeten betreiben.
Diese Problematik hat „Cäsar“ dazu veranlasst, eine Methode zu verwenden, mit der seine Nachrichten nur vom von ihm bestimmten Empfänger entschlüsselt und somit gelesen werden konnten. Sein Verfahren basiert auf der grundlegenden Idee *Buchstaben im Alphabet*[2] um einen Faktor „k“ *zu verschieben*, sodass jedem Buchstaben ein neuer zugewiesen wird. Wollte man z.B den Begriff „Verschluesselung“ nach seinem Verfahren für Unbefugte unlesbar machen, verschob man jeden einzelnen Buchstaben um z.B *drei Stellen* im Alphabet, sodass „A“ der neue Buchstabe „D“ zugewiesen wurde und erhielt daraus den verschlüsseltet Klartext „yhuvfkoxhvvhoxqj“. *Diese* Art der *Verschlüsselung*[3] *galt* zu Zeiten Cäsars *als sicher* und konnte mit einer beliebigen Verschiebung praktiziert werden.

---

[1] WasIstWas Tessloff Verlag URL: http://www.wasistwas.de/geschichte/eure-fragen/das-alte-rom/link//9ea729da91/article/wer-war-gaius-julius-caesar.html?tx_ttnews%5BbackPid%5D=1292 (Stand 26.02.2010)

[2] Dr. Michael Wagner URL: http://www.luk-korbmacher.de/Schule/Orga/se0117.htm (Stand: 23.02.2010)

[3] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Kryptographie (Stand: 23.02.2010)

1.2 Worin bestand die Problematik der ersten Verschlüsselungsmethoden?

Obwohl sich Cäsars Verschlüsselungsverfahren zunächst als sicher erwies, *fanden arabische Gelehrte im 9. Jahrhundert in der so genannten „Häufigkeitsanalyse“*[4] eine Möglichkeit, den verschlüsselten Text wiederum in den Ausgangsklartext zurückzuführen. Hierbei machte man sich die Besonderheit zu Nutze, dass bestimmte Buchstaben in verschiedenen Sprachen unterschiedliche Häufigkeiten aufweisen; *so lässt sich zum Beispiel der Buchstabe „E“ im Deutschen am häufigsten wiederfinden*. Da jedem „E“ in unserem Beispiel ein und derselbe Buchstabe „h“ zugeschrieben wird, lässt sich über einen verschlüsselten Text, der z.B am häufigsten ein „Z“ beinhaltet, aussagen, dass die Wahrscheinlichkeit, dass „Z“ dem Zeichen „E“ entspricht, sehr hoch ist und somit der Text eine Verschiebung von 21 Stellen aufweisen könnte. Mit dieser Methode war es somit innerhalb kurzer Zeit möglich, die Verschlüsselungsregelmäßigkeit zu erkennen, womit die Verschlüsselung aufgehoben war.

Ein weiteres Problem, was bedingt durch die angewandte Methode auftrat, war, dass nun zwar der Text nicht mehr in Klartext überliefert, dennoch aber *die angewandte Verschlüsselungsmethode dem Empfänger signalisiert* werden musste, was dadurch ein Sicherheitsrisiko hervorbrachte. Somit besteht die Schwäche und eigentliche Problematik dieser Verfahren im Transport des Geheimschlüssels, der zum Entschlüsseln der Nachricht notwendig ist.

---

[4] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Kryptographie (Stand: 23.02.2010)

1.3 Welche Veränderungen ergaben sich im Laufe der Neuzeit und was für Ziele verfolgte die moderne Kryptographie von dort an?

Mit dem *Bedeutungszuwachs des diplomatischen Briefverkehrs* und der zunehmenden Verbreitung des Schreibtelegrafen, welcher im Jahre *1844 von „Samuel Morse“*[5] konstruiert und dessen Übertragungen oftmals von Unbefugten abgehört wurden, stellte man neue Überlegungen zu grundlegenden Fragen der Kryptographie an. Hierbei stand vor allem die Problematik des Übertragungsweges im Vordergrund, wobei eine Verschlüsselung, deren Anwendung es erforderte, dem Empfänger Informationen über die verwendete Methode zukommen zu lassen, nicht ausreichend Sicherheit suggerierte. Um diesem entgegenzuwirken, entwickelte der niederländische *Kryptologe „Auguste Kerckhoffs von Nieuwenhof“* mit dem so genannten *„Kerckhoffs‘ Prinzip“* einen Verfahrensgrundsatz der Kryptographie, der besagt, dass „*die Sicherheit eines kryptographischen Verfahrens allein auf der Geheimhaltung des Schlüssels basieren soll*“. Diese Erkenntnis hat zu einem Wandel in der Methodik von Verschlüsselungsvorgängen geführt und eröffnete von nun an die Möglichkeit, das Kryptoverfahren an sich zu publizieren, während der Schlüssel für Unbefugte unzugänglich aufbewahrt wurde. Mit diesen grundlegenden Veränderungen kristallisierten sich „*vier Hauptziele zum Schutz von Informationen*“ heraus, die für die moderne Kryptographie auch heute von Bedeutung sind:
„Vertraulichkeit bzw. Zugriffsschutz“, „Integrität bzw. Änderungsschutz“, „Urheberschaft des Absenders“, „Verbindlichkeit bzw. Nichtabstreitbarkeit“
Diese vier Ziele zeigen verschiedene Aufgabenbereiche, in denen die Kryptographie zur Anwendung gelangt. Hierbei ist wichtig, dass ein „kryptographisches Verfahren“ nicht alle vier Absichten gleichzeitig verfolgen muss. In den nachfolgenden Ausführungen liegt das Hauptaugenmerk auf dem so genannten „Zugriffsschutz gegenüber Unbefugten“, weshalb dieses einer kurzen Klärung bedarf.
„Vertraulichkeit bzw. Zugriffsschutz“, befasst sich damit, Unbefugten den Zutritt zu sensiblen Daten zu versperren, wobei nur dem „Sender“ bzw. „Empfänger“ die benötigten Berechtigungen zugeschrieben werden, mit der Daten sowohl gelesen als auch, wenn die Erlaubnis hierfür vom Sender im Voraus erteilt worden ist, verändert werden können.

---

[5] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Telegrafie (Stand: 23.02.2010)

2. Entwicklung von symmetrischen hin zu asymmetrischen Kryptoverfahren

Im Wesentlichen lassen sich Verschlüsselungen zwei verschiedenen Sparten zuordnen, den „symmetrischen“ und den so genannten „asymmetrischen“ Kryptoverfahren. Beide besitzen ihre Vor- und Nachteile, weshalb immer genau differenziert werden muss, welche für den eigenen Anwendungszusammenhang in Frage kommt.

2.1 Symmetrische Verschlüsselungstechniken

2.1.1 Grundprinzip des symmetrischen Kryptoverfahrens

Im Allgemeinen unterscheiden sich symmetrische Kryptoverfahren von anderen dadurch, dass diese die Eigenschaft besitzen, *sowohl für den „Verschlüsselungs-“ als auch für den „Entschlüsselungsvorgang“ ein und denselben Schlüssel zu verwenden*[6], welcher mit „Private Key“ bezeichnet wird.
Möchte der Empfänger „E“ dem Sender „S“ eine Nachricht zukommen lassen, so müssen sich sowohl „E“ als auch „S“ auf einen „Private Key“ einigen, mit dem die Nachricht daraufhin „ver-“ bzw. „entschlüsselt“ wird.

2.1.2 Vorteile von symmetrischen Algorithmen

Zu den Vorteilen dieser Methode, Daten unkenntlich zu machen, gehört vor allem die Schnelligkeit; so können moderne Rechner eine Geschwindigkeit von rund *100Mb/s*[7] erreichen, sodass auch größere Datenmengen in einem möglichst geringen Zeitfenster verschlüsselt werden können. Ausserdem bieten symmetrische Kryptotechniken die Möglichkeit, dass sie *ohne Lizenzverletzungen frei zur Verfügung stehen*, weshalb sie jedem zugänglich sind und aufgrund dessen eine große Zielgruppe erreichen.

---

[6] URL: http://www.it-administrator.de/lexikon/symmetrische_verschluesselung.html (Stand 25.02.2010)

[7] URL: http://www.internet-sicherheit.de/institut/buch-sicher-im-internet/workshops-und-themen/verschluesselung-und-identitaeten/kryptographie/symmetrische-verschluesselungsverfahren/ (Stand 25.02.2010)

2.1.3 Schwächen dieser Art der Verschlüsselung und das daraus resultierende Bedürfnis nach einem weitreichenderen Verfahren

Zu den Nachteilen der Methode zählt die „Schlüsselvergabe“, wobei mit zunehmender Anzahl von Beteiligten, unter denen ein verschlüsselter Datenaustausch erfolgen soll, auch die Zahl an benötigten Schlüsseln deutlich steigt. Um die benötigte Schlüsselvielfalt für eine gegebene Teilnehmerzahl „n“ zu berechnen, wendet man die Formel „n*(n-1)/2“ an. So benötigt man zum Beispiel für „*100 Teilnehmer, 4950 Schlüssel“*, was das Verfahren zunehmend erschwert.
Ein weiterer Aspekt umfasst das erhöhte Sicherheitsrisiko, was sowohl bei der Schlüsselübergabe als auch bei der Aufbewahrung auftritt, wodurch insbesondere bei großen Teilnehmerzahlen eine *aufwändige und sichere Verwaltung* unverzichtbar erscheint.
Die Schwierigkeit dieser Verschlüsselungstechnik besteht darin, den geheimen Schlüssel zwischen „E“ und „S“ so auszutauschen, dass Unbefugte diesen nicht abfangen können, was zum Beispiel dadurch gewährleistet werden kann, dass der „Private Key“ *persönlich übergeben* wird.
Neben diesem Austausch bestehen Sicherheitsrisiken auch in der Aufbewahrung des jeweiligen Sicherheitsschlüssels. Kommt es dazu, dass dieser in die Hände unbefugter Personen gelangt, so lässt sich das Verschlüsselte leicht entschlüsseln. Demnach besteht ein großes Interesse nach einem weitreichenderen Verfahren, welches die Problematik des Schlüsseltransports in Angriff nimmt und dadurch eine höhere Sicherheit für den Anwender gewährleistet. Eine Lösung finden wir hierbei im so genannten „asymmetrischen Verschlüsselungsalgorithmus“.

2.2 Das Grundprinzip des „asymmetrischen Kryptoverfahren“

Um das grundlegende Verfahren zu erläutern, auf dem die asymmetrische Kryptografie basiert, möchte ich zunächst ein *Beispiel*[8] heranziehen:
Zwei natürliche Personen, „Peter“ und „Anna“ möchten untereinander einen Austausch einer geheimen Nachricht vollziehen, wobei „Peter“ die Position des Senders und „Anna“ die des „Empfängers“ einnimmt. Ihr Problem dabei, setzt sich daraus zusammen, dass der Transportweg von Spionen besetzt ist, weshalb die Nachricht schnell in fremde Hände geraten könnte. Aufgrund dessen ist es nicht ratsam, ein „symmetrisches Verfahren“ anzuwenden, da bei diesem der „Schlüssel“ übertragen und dadurch abgefangen werden könnte. Somit muss ein erweitertes Verfahren Anwendung finden, welches ohne einen direkten Schlüsselaustausch funktioniert.
Peters Idee besteht nun darin, Anna eine durch ein Schloss versperrte Kiste mit dem gewünschten Inhalt zukommen zu lassen, wobei Anna diese wiederum nach deren Erhalt mit einem eigenen Schloss versieht und diese an Peter zurückschickt. Nun entfernt Peter sein Schloss und schickt die Kiste ein letztes Mal an Anna, welche die Kiste nun durch die Entfernung des eigenen Schlosses öffnen kann. Auf diese Weise haben die Spione keine Chance und der sichere Datenaustausch ist somit erfolgt.
Die im Beispiel beschriebene Vorgehensweise ist der grundlegende Gedanke von „asymmetrischen Verfahren“, wobei der mehrfache Transport, wie er im Beispiel auftritt, nicht mehr erforderlich ist.

2.2.1 „Vor-“ und. „Nachteile“ von „asymmetrischen Kryptoverfahren“

Obwohl „asymmetrische“ Verfahren gegenüber „symmetrischen“ viele Vorteile aufweisen, bringen sie auch einige Nachteile, über die man sich vor dem eigentlichen Gebrauch in Kenntnis setzen muss, um die Methodik sicher anwenden zu können.

Zum bedeutendsten Vorteil zählt zunächst das *weitaus höhere Maß an Sicherheit*[9], was dadurch gewährleistet wird, dass „*der Private Key beim Empfänger verbleibt*“ und dadurch das Geheimnis „*nur von einer Person getragen werden muss*“.

---

[8] Katrin Schäfer, URL: http://www.matheprisma.uni-wuppertal.de/Module/RSA/index.htm Seite (Stand 26.02.2010)

[9] Philipp Hauer URL: http://www.philipphauer.de/info/info/asymmetrische-verschluesselung/ (Stand 26.02.2010)

Außerdem *kann* der „Public Key“, der zum Verschlüsseln der Nachricht benötigt wird, *veröffentlicht werden*, weshalb an dieser Stelle keine Sicherheitsrisiken auftreten können. Des Weiteren *kann der „Private Key“ nicht ohne weiteres berechnet werden*, da dies einen enormen Zeitaufwand, auch bei Verwendung leistungsstarker Rechner, bedeuten würde. In der heutigen Zeit des 21. Jahrhunderts finden „Private Keys“ von rund „*300 Stellen*“ Verwendung, wobei es bislang nur gelang, „*Schlüssel von rund 193 Stellen*“ zu brechen, was „einen Zeitraum von rund *einem Jahr* in Anspruch nahm“. Ein letzter Vorteil bezieht sich auf die *Anzahl der benötigten Schlüssel*. Im Gegensatz zu „symmetrischen Verfahren“ steigt diese *nicht exponentiell, sondern linear*, was bedeutet, dass die Anzahl proportional zur vorhandenen Teilnehmerzahl zunimmt. Dieser Aspekt erleichtert den Arbeitsaufwand bei einer Vielzahl von Teilnehmern erheblich.
Zu den Nachteilen zählt zunächst ein stark *gestiegener Rechenaufwand*. Im Vergleich zur „symmetrischen“ ist die „asymmetrische Verschlüsselung *ca. 1000 Mal langsamer*“, sodass sich eine Verlangsamung vor allem bei großen Datenmengen zeigt. Außerdem muss die Nachricht, sollte es mehrere Empfänger geben, mit jedem „Public Key“ der jeweiligen „Empfänger“ verschlüsselt werden, was einen „erhöhten Aufwand“ für den Sender bedeutet.

2.2.2 Führt die Kombination aus symmetrischen und asymmetrischen Verschlüsselungsverfahren zum Erfolg?

Da die zuvor erläuterten Verfahren neben Vorteilen auch Nachteile besitzen, ist die Frage berechtigt, ob die Kombination aus Beiden den gewünschten Erfolg erzielt. Hierbei lässt sich festhalten, dass tatsächlich beide Verfahren durch die Verbindung ihrer Haupteigenschaften, *erhöhte Sicherheit und weniger Rechenaufwand*, eine Möglichkeit bieten, große Datenmengen in kurzer Zeit zu verschlüsseln. Hierbei werden die Daten mit dem symmetrischen Verfahren verschleiert und der daraus erzielte Geheimschlüssel mit Hilfe der asymmetrischen Kryptotechnik übertragen.

3. RSA Verschlüsselungsverfahren

Die „RSA-Verschlüsselung“ ist ein Kryptoverfahren“, welches besonders im 21. Jahrhundert zur Anwendung gelangt. Es basiert auf den Grundlagen „asymmetrischer Kryptotechniken“ und *verwendet demnach sowohl einen „Public Key“*[10], welcher für die Verschlüsselung verwendet wird *und einen „Private Key“* mit dem sich der Geheimtext wiederum auf den Originalklartext zurückführen lässt. Die Sicherheit dieser Methodik *beruht auf der Annahme, dass es sehr schwer ist, aus dem Produkt „N“ zweier Primzahlen „p“,“q“, welches leicht durch „N=p*q“ berechnet werden kann, auf die „Ausgangsfaktoren“ zu schließen.*

3.1 Mathematische Grundlagen für das Verständnis von RSA:

3.1.1 Was sind Primzahlen und wie lassen sich diese berechnen?

Möchte man das RSA-Verfahren anwenden, so wird man feststellen, dass Primzahlen einen wesentlichen Bestandteil der Methode darstellen. Hierbei macht man sich deren Eigenschaft zu Nutze *genau zwei Teiler zu haben*[11]. So sind zum Beispiel die Zahlen „2,3,5,7,11,13“ Primzahlen, da sie das zuvor genannte Merkmal erfüllen.
Um Primzahlen zu berechnen, kann man zum Beispiel vom so genannten *„Sieb des Eratosthenes“, ein griechischer Mathematiker um 284 bis 202 v. Chr.*, Gebrauch machen. Wenn man alle Primzahlen von „2“ bis „200“ herausfinden möchte, schreibt man zunächst alle Zahlen in eine Reihe auf und streicht daraufhin alle Vielfachen von „2“, „3“, „5“, „7“, „11“, „13“, *bis zur Wurzel aus der gewünschten Obergrenze*, weg. Übrig bleiben nun alle Zahlen die genau zwei Teiler besitzen und somit die Kriterien für eine Primzahl erfüllen.

---

[10] Christian Vollmer URL: http://www.gierhardt.de/informatik/krypto/rsavollmer.pdf (Stand 28.02.2010)

[11] Robert Müller „Mathematik verständlich“, Seite 40-41

3.1.2 Einwegfunktionen / Einwegfunktionen mit Falltür

Einwegfunktionen zeichnen sich dadurch aus, dass es zwar „*ein effizientes Verfahren zu Berechnung*“ von „f(x)=y“, nicht aber für deren Umkehrung „x=f-1(y)“ gibt. So lässt sich die Funktion „f(x)=y“ in Sekunden berechnen, während deren Umkehrung mehrere Monate bis Jahre benötigen kann.
„*Beispielsweise*[12] *lässt sich aus Zutaten leicht Coca Cola herstellen; allerdings erweist es sich äußerst schwierig, aus dem fertigen Produkt zurück auf die Zutaten zu schließen.*“

3.1.3 Die Modulo-Rechung

Mit Hilfe der „Modulo-Rechnung“[13] lassen sich Reste aus Divisionen berechnen, wobei diese mit dem Zeichen [mod] abgekürzt wird. Bei einer Division von 37/8 berechnet sich der Rest zum Beispiel mit „37 mod 8= 5“, denn „4*8=32“; Rest 5.

3.1.4 Die eulerische Funktion

Die „Eulerische Funktion“ der Form L = (p-1)*(q-1), gibt an „*wie viele natürliche Zahlen es gibt, die zu „p“ und „q“ teilerfremd sind*“, was bedeutet, dass sie keinen gemeinsamen Teiler besitzen.

---

[12] Lukas Dölle URL: http://www2.informatik.hu-berlin.de/Forschung_Lehre/algorithmenII/Lehre/WS2002-2003/Perlen/Variationen.pdf (Stand 27.02.2010)

[13] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Division_mit_Rest (Stand 26.02.2010)

4.1 Projekt - Das Erstellen eines eigenständigen Programms für die Verschlüsselung einer zuvor gegebenen Nachricht

Nachdem nun die für das Verständnis des Algorithmus erforderlichen mathematischen Grundlagen geklärt sind, möchte ich dazu übergehen, meine Programmidee zu erläutern.
Im Mittelpunkt meiner Überlegungen stand das Interesse herauszufinden, ob es mir, mit Hilfe der im Zeitraum der Facharbeit erlangten Kenntnisse über das RSA-Verfahren, möglich ist, ein eigenes Programm zu schreiben, welches dem Anwender die Möglichkeit gibt, sowohl Nachrichten zu „ver-“ als auch den dadurch erlangten Geheimtext durch den gewünschten Empfänger wieder zu entschlüsseln. Folglich muss das gewünschte Programm in zwei voneinander unabhängige Teile separiert werden, wobei der Erste die Aufgabe der Schlüsselerzeugung und der Zweite die der „Ver-“ bzw. „Entschlüsselung“ übernimmt.
Während des Programmierungsprozesses war es mir vor allem wichtig, die einzeln aufeinander aufbauenden Schritte so zu optimieren, dass das Endresultat ressourcenarm, schnell und benutzerfreundlich zu bedienen ist. In den nachstehenden Erläuterungen möchte ich mein Vorgehen während der Programmentwicklung beschreiben, auf auftretende Probleme eingehen und gleichzeitig die Funktionsweise von RSA verständlich machen. Hierbei soll die Nachricht „Die RSA Verschlüsselung ist sicher!“ als Anwendungsbeispiel ver- bzw. entschlüsselt werden.

4.1.1 Schritt 1: Produktbildung „N“ der beiden Primzahlen „p“ und „q“

Zunächst müssen *zwei Primzahlen „p“,“q“ ermittelt werden*[14], für die gilt p $\neq$ q.
Schon an dieser Stelle zeigte sich das erste Problem, was daraus bestand, dass „Revolution“[15] keine Funktion zum bestimmen von beliebig großen Primzahlen zur Verfügung stellt, weshalb ich mir ein eigenes Verfahren überlegen musste, welches diese Aufgabe übernimmt. Ich überlegte mir, eine Funktion zu schreiben, die mir alle Primzahlen bis zu einer festgelegten Obergrenze berechnet. Um dies zu realisieren,

---

[14] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/RSA-Kryptosystem Stand (25.02.2010)

[15] verwendete Programmierungssoftware URL: http://www.runrev.com/ (Stand 1.03.2010)

müssen zwei „Schleifen“ miteinander kombiniert werden, mit Hilfe derer alle Zahlen von 2 bis zur gegebenen Obergrenze auf ihre Teilbarkeit geprüft werden:
Die gefundenen Primzahlen werden hintereinander, mit Komma getrennt, in ein Textfeld geschrieben. Nun sollen zwei beliebige Primzahlen aus dem Feld herausgenommen werden, was zufällig geschehen soll. Mit der so genannten „random (z) Funktion“, wobei „z“ die Anzahl der Items innerhalb des Feldes darstellt, wird über den Button „Zufallsprimzahlen n,e;d“ ein Zufallsitem bestimmt, welches daraufhin in ein weiteres Textfeld „p1“ geschrieben wird. Da diese gefundene Primzahl nicht mehr zur Verfügung steht, wird sie aus dem Primzahlenfeld gelöscht, sodass die Zweite beliebig gefundene Primzahl „p2“ nicht gleich der Ersten sein kann. Automatisch wird nun das Produkt aus „p1“ und „p2“ gebildet und in das Textfeld „N“ geschrieben. Sei „p1= 7“ und „p2= 11“, so wäre das Produkt aus beiden,“N= 77“. Dieses „N“ bildet mit „e“ den „Public Key“ und mit „d“ den „Private Key“.

4.1.2 Schritt 2: Erzeugen der Schlüssel

Im 2. Schritt müssen nun die beiden übrigen Bestandteile der Schlüssel neben „N“ bestimmt werden. Zunächst berechnen wir „e“, welches eine beliebige Zahl sein kann, *die jedoch mit L= (p-1)*(q-1) keinen gemeinsamen Teiler haben darf*[16]. Dazu suchen wir uns eine weitere Zufallszahl, zum Beispiel „23“, und überprüfen diese darauf, ob sie die Bedingung teilerfremd zu L= (p-1)*(q-1) zu sein, erfüllt. Ist dies der Fall, wird sie in das dafür vorgesehene Feld „p3“ geschrieben, ist dies nicht der Fall, wird derselbe Vorgang innerhalb der Schleife mit einer anderen Zufallszahl durchgeführt. In unserem Beispiel („N= 77, p1= 7“ und „p2= 11“) wäre „L= 60“. Demnach wäre der „Public Key“ von zum Beispiel (N=77,e=23) gefunden.
Das Berechnen des „Private Key“ (N,d) erwies sich als deutlich schwieriger und konnte nur durch den „*erweiterten euklidischen Algorithmus*“[17] bestimmt werden. Um „d“ zu berechnen, wendet man die Gleichung „ $e * d = 1 + L * t$ “[18] an, wobei „e“ und „L“ schon bekannt sind und *„t“ für ein beliebig Vielfaches von „L“* steht.

---

[16] FH Flensburg URL: http://www.iti.fh-flensburg.de/lang/krypto/rsa.htm (Stand: 13.02.2010)

[17] Programmierungshilfe URL: http://www2-fs.informatik.uni-tuebingen.de/~reinhard/krypto/German/2.2.d.html (Stand 23.02.2010)

[18] „Creative Commons Attribution/Share Alike“ Verständnishilfe URL: http://de.wikipedia.org/wiki/Erweiterter_euklidischer_Algorithmus (Stand 23.02.2010)

Der „erweiterte euklidische Algorithmus“ berechnet neben dem größten gemeinsamen Teiler, zweier Zahlen, *auch zwei ganze Zahlen „d“ und „t“*, die die folgende Gleichung erfüllen“: $ggT(e,L) = d * e + t * L$.

Das gesuchte „d“ kann nun ohne weiteres abgelesen und ins Textfeld „d“ geschrieben werden; „t“ wird nicht weiter benötigt. In unserem Fall ist „d = 47“.

Die Berechnung des „Private Key“ in „Revolution“ zu programmieren hat mich viel Zeit und Mühe gekostet, da dieser Vorgang sehr komplex und schwierig erschien. Als Hilfestellung, um die Rechnung die der Computer vornimmt, nachvollziehbar zu gestalten, habe ich mich dazu entschlossen, ein Textfeld zu erstellen, in dem die Berechnung von „d“, abgestimmt auf die gegebenen Werte (L,e), aufgeschrieben wird und somit nachvollzogen werden kann.

4.1.3 Schritt 3: Das eigentliche Verschlüsseln der Nachricht

Nun, wo sowohl „Public - “ als auch „Private - Key“ gefunden sind, können wir zum Verschlüsseln der Nachricht übergehen. Die Verschlüsselung der Nachricht erfolgt durch die Formel „$C = M^e \bmod N$“. Hierbei entspricht „N,e“ (77,23) dem öffentlichen Schlüssel, „M“ der Nachricht die verschlüsselt werden soll, in unserem Beispiel „Die RSA Verschlüsselung ist sicher!“ und „C“ dem Geheimtext, den wir erhalten möchten. Das Verfahren beruht nun auf der Annahme, dass „$(M^e)^d \bmod N = M$“ ergibt, was im Nachfolgenden noch zu beweisen ist. Somit lässt sich das verschlüsselte „C“ mit Hilfe der Falltür „d“ entschlüsseln, sodass „$C^d \bmod N = M$“ entspricht.

Da man nicht mit Buchstaben rechnen kann, müssen wir jedem Zeichen eine bestimmte Zahl zuordnen, was zum Beispiel dadurch erfolgen kann, dass wir den Klartext in „ASCII“ umkonvertieren lassen. *„ASCII“*, was für *„American Standard Code for Information Interchange“*[19] steht, wurde *1963 entwickelt* und umfasst eine Zuordnungsfestlegung, wobei insgesamt *128 Zeichen* einen Wert erhalten, der in unserem Fall dazu beiträgt, mit „Zeichen rechnen zu können“. Aufgrund der Tatsache, dass sich „ASCII“ schnell etablierte, *beherrschen heutzutage fast alle Rechner diese Zuordnung*, weshalb der entsprechende Wert leicht von anderen Systemen verstanden werden kann.

---

[19] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Ascii (Stand 24.02.2010)

In Bezug auf mein Programm habe ich mit der Funktion „chartonum ()“ jedem Zeichen den entsprechenden „ASCII- Wert“ zugeordnet und diese, durch Komma getrennt, in ein weiteres Textfeld mit dem Namen „ASCII“ ausgegeben. So entspricht unser Beispieltext den folgenden Werten: („68,105,101,32,82,83,65,32,86,101,114,115,99,104,108,159,115,115,101,108,117,110, 103,32,105,115,116,32,115,105,99,104,101,114,33“), welche nun mittels der zuvor genannten Formel verschlüsselt werden können.

Da bei diesem Schritt mit großen Zahlen potenziert werden muss, stellte sich nach einigen Versuchen heraus, dass die Zahlen zu groß wurden und demnach ein Programmfehler auftrat. Ich musste eine Methode finden, die es mir ermöglichte, in relativ kurzer Zeit große Zahlen potenzieren zu können. Schließlich wurde ich in der Methode, die den Namen „*schnelles Potenzieren*“[20] trägt, fündig. Möchte man „x^e“ berechnen, wobei „x“ für die Basis und „e“ für den Exponenten steht, so müsste man „x“ bei einem „e = 23“ dreiundzwanzig Mal mit sich selbst multiplizieren. Dies bedeutet vor allem bei großen Exponenten einen *hohen Rechenaufwand*. Beim „schnellen Potenzieren“ macht man sich die *Binärdarstellung* von „e“ zu Nutze. In unserem Beispiel sähe das wie folgt aus:

e	512	256	128	64	32	16	8	4	2	1
23	0	0	0	0	0	1	0	1	1	1
oder z.B 624	1	0	0	1	1	1	0	0	0	0

„23“ entspricht somit der Binärdarstellung „10111“ und „624“ „1001110000“.

Nun muss man für jede „1“ „QM“ und für jede „0“ „Q“ setzen. So ergibt sich bei „e=23“ die Reihe „QM,Q,QM,QM,QM“, *wobei immer das erste Paar weggelassen wird*. So erhält man „Q,QM,QM,QM“. Diese Reihe nutzen wir nun als Vorgabe für die anzuwendenden Rechenoperationen. Möchten wir den ersten Buchstaben von „Die RSA Verschlüsselung ist sicher!“, ein „D“ bzw. „68“ (in ASCII) verschlüsseln, so müssen wir 68^{23} rechnen, *indem wir nach „Q,QM,QM,QM“ vorgehen* und bei jedem „Q“ das Zwischenergebnis quadrieren und bei jedem „M“ mit der Basis multiplizieren. Auf diese Weise sparen wir von den „22“ Schritten, die wir eigentlich benötigt hätten, „15“,

[20] „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Binäre_Exponentiation (Stand 24.02.2010)

sodass „7“ Rechenschritte übrig bleiben. „$(((((((68^2)^2)*68)^2)*68)^2)*68)$“ liefert uns das Ergebnis: $1,405092748*10^{42}$. Nach unserer Verschlüsselungsformel („$C = M^e \bmod N$“) müssen wir nun noch „modulo N“ vollziehen. Somit erhalten wir für „D“ den Geheimtext „52“.

4.1.4 Schritt 4: Entschlüsseln

Das Entschlüsseln des Geheimtextes „C“ zeigt sich ähnlich wie das Verschlüsseln. Hierbei finden dieselben Rechenoperationen statt, wobei lediglich der Exponent nicht mehr „e“ sondern der Falltür „d“ entspricht. Über „$C^d \bmod N = M$“ berechnen wir wiederum mit Hilfe des *schnellen Potenzierens* „52^{47}“. Das Ergebnis wird daraufhin noch „modulo N(60)“ genommen, sodass wir schließlich wiederum die „ASCII-Schreibweise“ „68“ unseres verschlüsselten Buchstaben „D“ erhalten. Diese Prozedur wird nun in einer Schleife, die genauso lange andauert, wie es Buchstaben gibt, abgehandelt und liefert uns schließlich den entschlüsselten Klartext in das Textfeld „Entschlüsselt“.
An dieser Stelle haben wir es nun geschafft sowohl „Public - “ als auch „Private - Key“ zu generieren, eine Nachricht daraufhin mit den entsprechenden Schlüsseln zu codieren und schließlich diese wiederum mittels der Falltür „d“ auf den Ausgangsklartext zurückzuführen.

4.2 Beweis: Warum das „RSA - Verfahren“ funktioniert

Aufgrund meiner Teilnahme bei der „Schülerkrypto 2010“ in Bonn, erhielt ich die Möglichkeit, eine Studentin nach einigen Verständnisschwierigkeiten, die sich mir insbesondere in Bezug auf den Beweis des RSA-Verfahren ergaben, zu befragen. Aufgrund dessen möchte ich nun versuchen, meine neue erlangten Kenntnisse in Form des Beweises verständlich darzulegen:
Die nachfolgenden Überlegungen basieren auf dem „Satz von Fermat“ und dem „Satz von Lemma“.

4.2.1 Satz von Fermat[21]

Sei „p“ eine beliebige Primzahl und „x“ nicht teilbar durch „p“, so gilt:

$\Rightarrow x^{p-1} \equiv_N x$

4.2.2 Satz von Lemma[22]

Seien „p“ und„q“ zwei beliebige Primzahlen und „p“ ≠„q“ und „x“, „y“ $\in R$, so gilt:

Wenn $(x \equiv_p Y)$ und $(x \equiv_q Y)$ dann ist auch $x \equiv_{p*q} Y$

4.2.3 Beweis: Satz von Lemma[23]

Als Voraussetzung für den Satz von Lemma muss „p“ und auch „q“ ein Teiler von „(x-y)“ sein. In Bezug auf die „Primfaktorzerlegung“ von „(x-y)“ lässt sich feststellen, dass „ sowohl „p“ als auch „q“ in dieser vorkommen, weshalb „(x-y)“ ein Vielfaches von „p*q“ sein muss. Aus $x - y = f * (p * q)$ folgt $x \equiv_{p*q} y$.

Verallgemeinert man nun den Satz von Fermat auf zwei Primzahlen so besagt er, dass $x^{(p-1)*(q-1)} \equiv_{p*q} 1$ gilt, was die folgenden Termumformungen beweisen:

$x^{(p-1)*(q-1)} = p * (x^{(p-1)})^{(q-1)} = p^{1^{(q-1)}} = p = 1$. Diese sind genauso für „q“ durchzuführen.

4.2.4 RSA - Beweis

Mein Ziel ist es nun, zu zeigen, dass $x^{e*d} \equiv_N x$, was auch als $x^{e*d} = x \bmod N$ geschrieben werden kann, gilt, was durch Einsetzen der sowohl „Ent-“ als auch „Verschlüsselungsformel“ $C = M^e \bmod N$ bzw. $C^d \bmod N = M$ ineinander erzielt werden kann. Zunächst gelten für das „RSA - Verfahren“ die grundlegenden Formeln „L= (p-1)*(q-1)“ bzw, „N=p*q“. Außerdem gilt für die Falltür „d“, dass „e*d -1 teilbar durch L“ sein muss, was wie folgt hergeleitet werden kann:

„$d * e = 1 \bmod L \mid -1 \Leftrightarrow d * e - 1 = 0 \bmod L$ “

---

[21] Der Beweis des Satz von Fermat erwies sich äußerst schwierig, weshalb dieser weggelassen wurde.

[22] Michael Nüsken „Warum RSA funktioniert“ - Informationsblatt

[23] Wiedergabe aufgrund von Aufzeichnungen während der Schülerkrypto 2010. Erklärung des Beweises durch Studentin „Christiane Beyer“.

Falls für „e“ und „d“ gilt: $e*d=1+L*t$ mit einem geeigneten „t“, so gilt auch:

$$\Rightarrow x^{e*d} \equiv_N x^{(1+L*t)} \equiv_N x^1 * x^{L*t}$$

$$\Leftrightarrow x^{e*d} \equiv_N x*(x^L)^t \longleftarrow \text{L= (p-1)*(q-1) einsetzen}$$

$$\Leftrightarrow x^{e*d} \equiv_N x*(x^{(p-1)*(q-1)})^t$$

$$\Leftrightarrow x^{e*d} \equiv_N x*1^t$$

$$\Leftrightarrow x^{e*d} \equiv_N x \leftarrow \text{Somit ist bewiesen, dass aus } x^{e*d} \text{ wiederum } x \bmod N \text{ folgt.}$$

5. Die Sicherheit des „RSA-Verfahren“ und die daraus resultierenden Zukunftsaussichten

An dieser Stelle, nachdem ich das „RSA-Verfahren“ anhand meines Programms erklärt habe, möchte ich mir zu guter Letzt die Frage stellen, wie sicher diese Verschlüsselungsmethodik in der heutigen Zeit, dem 21. Jahrhundert ist und ob es nicht sein kann, dass es eine Methode gibt, mit der sich der Algorithmus knacken lässt.
Wie ich in meiner Facharbeit festgestellt habe, beruht die Sicherheit des Algorithmus auf der *Schwierigkeit*[24] *die beiden Primzahlen „p“ „q“ aus dem Produkt „N“ beider zu bestimmen*. Obwohl Verfahren existieren, die es ermöglichen, die Faktoren aus beliebigen natürlichen Zahlen zu berechnen, zeigen sich diese als *sehr zeitaufwändig*, weshalb ein realistischer Angriff auf das angewendete „asymmetrische Verfahren“ aussichtslos erscheint. Dennoch ist zu beachten, dass es durchaus sein kann, dass in Zukunft ein *schneller Algorithmus zum Zerlegen von großen Zahlen in deren Primfaktoren, gefunden wird*, was zur Nichtigkeit von „RSA“ führen würde. Solange nicht bewiesen ist, dass es keinen Algorithmus zum Berechnung eben dieser Primzahlen gibt, bleibt ein gewisses Restrisiko bestehen.
Betrachtet man den Zeitaufwand bestehender „Faktorisierungsalgorithmen“, so lässt sich feststellen, dass deren Schnelligkeit vor allem *in Abhängigkeit zur „Hardwareentwicklung“* steht. In der vergangenen Zeit hat die Geschwindigkeit von Prozessoren immer weiter zugenommen, sodass es für das sichere Verschlüsseln zum jetzigen Zeitpunkt erforderlich ist, „Schlüssellängen“ zur Anwendung zu bringen, von

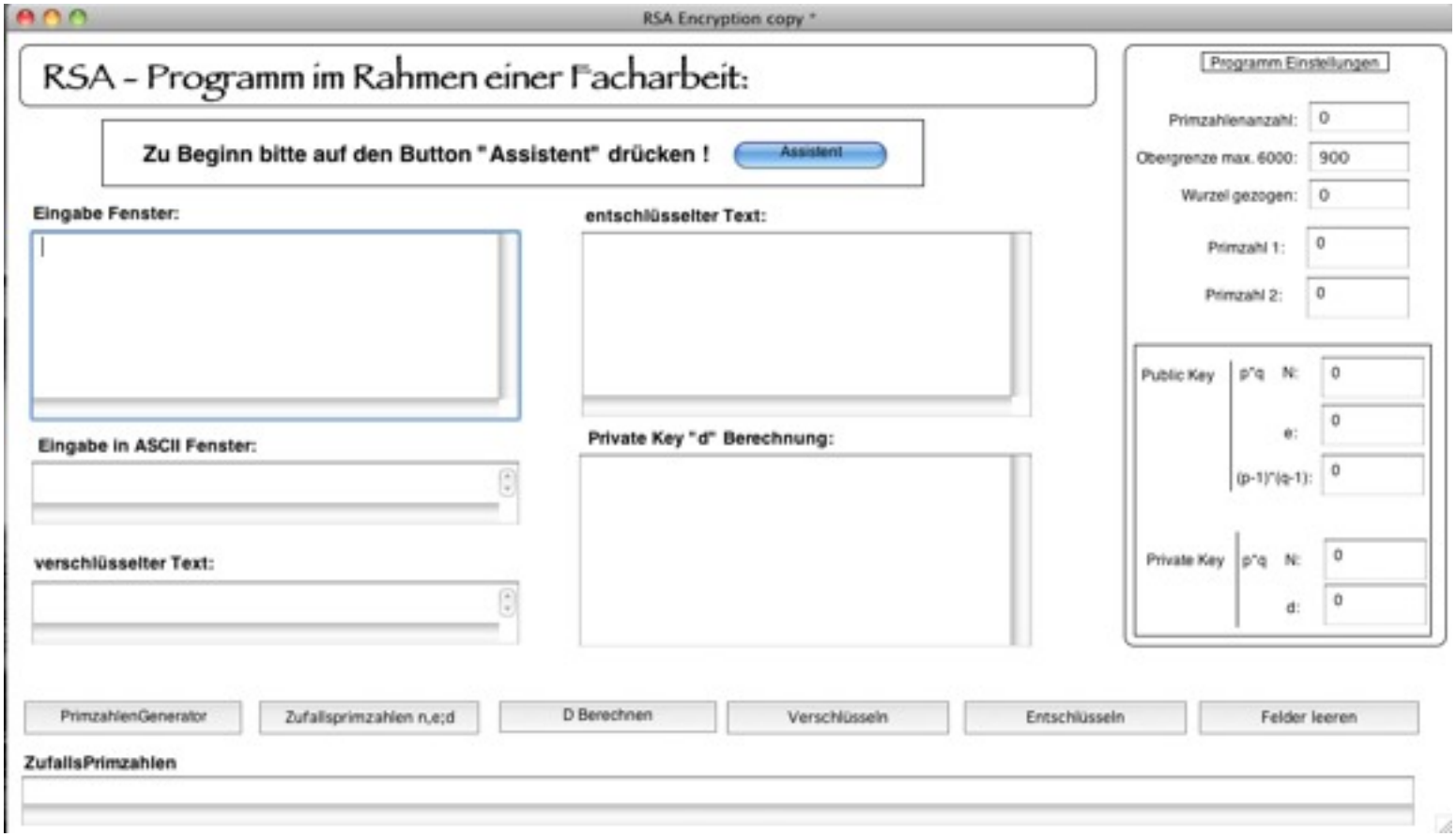
[24] Arno Wacker URL: http://www.iti.uni-stuttgart.de/~stankats/Buch/Buch_rsa.2.html (Stand 25.02.2010)

denen man annehmen kann, dass sie auch unter Berücksichtigung der Prozessorentwicklung, zukunftsorientierte Sicherheit gewährleisten. Möchte man heutzutage, im 21. Jahrhundert, einen Datensatz „sicher“ verschlüsseln, *so sollte man eine Schlüssellänge von 2048bit verwenden*, was allerdings nur auf Schätzungen beruht. Die nachstehende Tabelle soll verdeutlichen, welch enormer Zeitaufwand nötig ist, um verschiedene Schlüssellängen zu faktorisieren:

5.1 Fazit

Um meine Facharbeit abzuschließen, möchte ich zu einem Fazit gelangen, in dem ich nochmals einen kurzen Überblick darüber gebe, ob ich mein erwünschtes Ziel nach meiner Zufriedenheit erreicht bzw. wie ich den Verlauf meiner selbstständigen Arbeit, mit Schwerpunkt meines Projekts, erlebt habe.
Zusammenfassend hat sich gezeigt, dass es durchaus möglich ist ein eigenes Programm, mit den im Rahmen der Facharbeit erlangten Kenntnisse, zu programmieren, was die Möglichkeit eröffnet, beliebige Nachrichten zu ver- bzw. entschlüsseln.
Das eigenständige Arbeiten hat mir sehr viel Freude bereitet und ermöglichte mir einen tiefgründigen Einblick in die Welt der Kryptographie. Während des Programmierprozesses begegnete ich ständig neuen Herausforderungen, die es zu bewältigen galt. Oftmals konnte ich dabei nur auf englischsprachige Informationen zurückgreifen, was mir einen zusätzlichen Ansporn verlieh.
In Bezug auf mein Programm lässt sich sagen, dass die gefundene Lösung nicht optimal, aber dennoch für einen ersten Versuch zufriedenstellend ist. Eine Problematik, die sich mir noch stellte, war die Tatsache, dass in meiner Umsetzung jedes Zeichen einzeln verschlüsselt und im Anschluss in eine durch Komma getrennte Zahlenreihe geschrieben wird. Dies führt dazu, dass das Verfahren im Vergleich zu internationalen Standards keine absolute Sicherheit suggeriert, dennoch aber einen Einblick in den Anwendungszusammenhang mit „RSA“ ermöglicht.
Insgesamt bin ich der Meinung, die richtige Themenauswahl getroffen zu haben. Sowohl der Erfolg, selber ein eigenständiges Programm entwickelt, als auch die Möglichkeit, bei der „Schülerkrypto - Veranstaltung“ der Universität in Bonn Mathematik der besonderen Art erlebt zu haben, eröffneten mir die Möglichkeit, mich einer intensiven Auseinandersetzung mit Kryptographie zu widmen, welche aktuell und in Zukunft einen immer höheren Stellenwert einnimmt.

6. Endresultat



7. Literaturverzeichnis:

„Wer war Gaius Julius Cäsar ?“,WasIstWas Tessloff Verlag, URL: http://www.wasistwas.de/geschichte/eure-fragen/das-alte-rom/link//9ea729da91/article/wer-war-gaius-julius-caesar.html?tx_ttnews%5BbackPid%5D=1292 (Stand 26.02.2010)

„Kryptografie in der Praxis“, Dr. Michael Wagner, URL: http://www.luk-korbmacher.de/Schule/Orga/se0117.htm (Stand: 23.02.2010)

„Kryptographie“, „Creative Commons Attribution/Share Alike“, URL: http://de.wikipedia.org/wiki/Kryptographie (Stand: 23.02.2010)

„Telegrafie“, „Creative Commons Attribution/Share Alike“, URL: http://de.wikipedia.org/wiki/Telegrafie (Stand: 23.02.2010)

„Symmetrische Verschlüsselung“, URL: http://www.it-administrator.de/lexikon/symmetrische_verschluesselung.html (Stand 25.02.2010)

„Symmetrische Verschlüsselungsverfahren“, URL: http://www.internet-sicherheit.de/institut/buch-sicher-im-internet/workshops-und-themen/verschluesselung-und-identitaeten/kryptographie/symmetrische-verschluesselungsverfahren/ (Stand 25.02.2010)

„RSA Verschlüsselung“, Katrin Schäfer, URL: http://www.matheprisma.uni-wuppertal.de/Module/RSA/index.htm Seite (Stand 26.02.2010)

„asymmetrische Verschlüsselung“, Philipp Hauer URL: http://www.philipphauer.de/info/info/asymmetrische-verschluesselung/ (Stand 26.02.2010)

„Das RSA Verschlüsselungsverfahren“, Christian Vollmer URL: http://www.gierhardt.de/informatik/krypto/rsavollmer.pdf (Stand 28.02.2010)

Robert Müller, „Mathematik verständlich“, Seite 40-41

„Einwegfunktionen Variationen und Beispiele“Lukas Dölle URL: http://www2.informatik.hu-berlin.de/Forschung_Lehre/algorithmenII/Lehre/WS2002-2003/Perlen/Variationen.pdf (Stand 27.02.2010)

„Division mit Rest“, „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Division_mit_Rest (Stand 26.02.2010)

„RSA Kryptosystem“, „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/RSA-Kryptosystem Stand (25.02.2010)

„Runtime Revolution“, verwendete Programmierungssoftware URL: http://www.runrev.com/ (Stand 1.03.2010)

„RSA-Verfahren“, FH Flensburg URL: http://www.iti.fh-flensburg.de/lang/krypto/rsa.htm (Stand: 13.02.2010)

„Erweiterter Euklidischer Algorithmus“, URL: http://www2-fs.informatik.uni-tuebingen.de/~reinhard/krypto/German/2.2.d.html (Stand 23.02.2010)

„Erweiterter Euklidischer Algorithmus“, „Creative Commons Attribution/Share Alike“, URL: http://de.wikipedia.org/wiki/Erweiterter_euklidischer_Algorithmus (Stand 23.02.2010)

„ASCII“, „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Ascii (Stand 24.02.2010)

„Binäre Exponentation“, „Creative Commons Attribution/Share Alike“ URL: http://de.wikipedia.org/wiki/Binäre_Exponentiation (Stand 24.02.2010)

Michael Nüsken, „Warum RSA funktioniert“ - Informationsblatt

„Das RSA-Verfahren“,Arno Wacker URL: http://www.iti.uni-stuttgart.de/~stankats/Buch/Buch_rsa.2.html (Stand 25.02.2010)

Wiedergabe aufgrund von Aufzeichnungen während der Schülerkrypto 2010. Erklärung des Beweises durch Studentin „Christiane Beyer“.

weitere Nebenquellen:

„Klassische Krypthografie“, URL: http://www.cs.uni-potsdam.de/ti/lehre/04-Kryptographie/slides/EinfacheKryptosysteme.pdf, (Stand 20.02.2010)

„Asymmetrische Verschlüsselungsverfahren“, URL: http://www.mathematik.de/ger/information/wasistmathematik/rsa/rsa.html?print=1, (Stand 20.02.2010)

„Die RSA-Verschlüsselung“, URL: http://www.edi84.de/rsa/Die_RSA-Verschluesselung.html (Stand 20.02.2010)

„Modular Exponentation“, Universität Tuebingen, URL: http://www2-fs.informatik.uni-tuebingen.de/~reinhard/krypto/English/english.html, (Stand 18.02.2010)

8. Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die im Literaturverzeichnis angegebenen Hilfsmittel verwendet habe. Insbesondere versichere ich, dass ich alle wörtlich und sinngemäßen Übernahmen aus anderen Werken nach bestem Gewissen als solche kenntlich gemacht habe.

Elsdorf, den 03.03.2010 _____