Untersuchung der Anwendung homomorpher Kryptosysteme unter Berücksichtigung der unautorisierten Verformbarkeit von Chiffretexten

Matthias Ulbrich
2743974

Bachelorarbeit

Erstprüfer: Prof. Dr. Michael Meier
Zweitprüferin: Jun.-Prof. Dr.-Ing. Delphine Reinhardt
Betreuerin: Dipl.-Inform. Saffija Kasem-Madani

Institut für Informatik IV
Arbeitsgruppe für IT-Sicherheit
Rheinische Friedrich-Wilhelms-Universität Bonn

Danksagung

Besonders bedanken möchte ich mich bei meiner Betreuerin Dipl.-Inform. Saffija Kasem-Madani für die intensive akademische Begleitung während der Bachelorarbeit und bei Prof. Dr. Michael Meier für die Gelegenheit in seiner Arbeitsgruppe die Bachelorarbeit zu schreiben.

Weiterer Dank gilt meinen Eltern, meiner Partnerin, meiner Schwester und meiner Großtante für die große Unterstützung in meinem Studium bis zum jetzigen Zeitpunkt. Ich widme euch diese Arbeit.

Selbstständigkeitserklärung

Hiermit versichere ich, die vorliegende Bachelorarbeit ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Bonn, 5. Juli 2017

Matthias Ulbrich

Zusammenfassung

Diese Bachelorarbeit untersucht den vielfältigen Einsatz homomorpher Kryptosysteme in Forschungsarbeiten. Dabei wird erörtert welches Kryptosystem zum Einsatz kommt und aus welchen Gründen sich für dieses entschieden wurde. Es wird untersucht in welchem Anwendungskontext das Kryptosystem zum Einsatz kommt und ob eine unautorisierte Verformbarkeit der Chiffretexte (engl. *malleability*) bei der Implementierung berücksichtigt wurde. Unter der Verformbarkeit von Chiffretexten versteht man die Möglichkeit eines Angreifers mit verschlüsselten Zahlen eine Funktion wie die Addition zweier Zahlen auszuführen. Normalerweise würde ein Angreifer dazu beide Zahlen zunächst separat entschlüsseln müssen, wofür er einen geheimen Schlüssel als Eingabe für den Entschlüsselungsalgorithmus benötigt. Dann führt der Angreifer die Addition auf den Klartexten durch und verschlüsselt ihr Ergebnis wieder. Homomorphe Kryptosysteme ermöglichen Berechnungen mit verschlüsselten Zahlen wie z.B. deren Addition ohne Kenntnis eines privaten Schlüssels und ohne vorherige Entschlüsselung. Damit ist ein Angreifer bei einem homomorphen Kryptosystem in der Lage verschlüsselte Zahlen zu verändern. Findet diese Veränderung unbemerkt und unerlaubt statt, wird die Integrität von Rechenoperationen im Chiffreraum verletzt.

Die evaluierten Einsatzszenarien homomorpher Kryptosysteme werden im Einzelnen kurz vorgestellt um anschließend Kriterien zu erstellen anhand welcher eine Kategorisierung der Kryptosysteme erfolgt. Dies geschieht im Hinblick darauf Dritte bei der Entscheidung für den Einsatz homomorpher Kryptographie in eigenen Arbeiten zu unterstützen.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

In diesem Kapitel wird die inhaltliche Ausrichtung der Bachelorarbeit vorgestellt.

Ziel dieser Arbeit ist eine Erörterung der Vorteile von homomorphen Kryptosystemen für praktische Anwendungen. Homomorphe Kryptographie ermöglicht das Rechnen mit verschlüsselten Zahlen ohne diese selber zu kennen. Die homomorphen Kryptosysteme, welche in dieser Arbeit behandelt werden, ermöglichen Rechenoperationen mit verschlüsselten Zahlen und genau einem mathematischen Operator. Man nennt solche Kryptosysteme semihomomorph. Während semihomomorphe Kryptosysteme nur eine mathematische Operation auf Chiffren ermöglichen - abhängig von dem gewählten Kryptosystem - erlauben vollhomomorphe Kryptosysteme die Berechnung bis hin zu jeder booleschen Funktion im Chiffreraum im Falle von Gentry [Gen09]. Da letztere jedoch aufgrund hoher Anforderungen an Laufzeiten und Speicherverbrauch weniger praktikabel sind, greift man in der Praxis auf semihomomorphe Kryptosysteme zurück. Dies führt zu verschiedenen, vom jeweiligen Anwendungsfall abhängigen Einsatz eines Kryptosystems.

1.1 Beitrag dieser Bachelorarbeit

Die semihomomorphen Kryptosysteme unterscheiden sich nicht nur in der mathematischen Operation, die Sie für das Rechnen mit verschlüsselten Zahlen ermöglichen. Für die gleiche Operation existieren verschiedene semihomomorphe Kryptosysteme. In dieser Arbeit sollen die Auswahlkriterien für das eingesetzte Kryptosystem identifiziert werden. Die gesammelten Erkenntnisse werden anschließend benutzt um die einzelnen Kryptosysteme zu klassifizieren als Referenz für den zukünftigen Einsatz in Forschungsarbeiten.

Der Kompromiss zwischen Sicherheit und Praktikabilität ist ein wiederkehrendes Thema in der Kryptographie. Der Gewinn an Praktikabilität durch den Einsatz homomorpher Kryptosysteme wird mit eingeschränkter Sicherheit wegen der unautorisierten Verformbarkeit von Chiffretexten durch unautorisierte Dritte erkauft. Das Rechnen im Chiffreraum kann somit auch als Schwäche des Kryptosystems ausgelegt werden, da beliebige Dritte in der Lage sind verschlüsselte Information zu verändern und so die Integrität der hinterlegten Information anzugreifen. Bei der Analyse der Anwendungsfälle wird untersucht wie mit der Verformbarkeit von Chiffretexten durch einen Angreifer im Einzelnen Umgegangen wird um eine Integrität der hinterlegten Daten zu sichern und das Abschöpfen von Informationen aus dem System zu verhindern.

1.2 Aufbau

Zunächst werden im Kapitel 2 der Begriff eines Kryptosystems und darauf aufbauende Erweiterungen bis hin zum probabilistischen asymmetrischen Kryptosystem erläutert, welches die Grund-

lage vieler semihomomorphen Verfahren bildet. Dann werden in Kapitel 3 verschiedene Sicherheitskriterien erläutert, die in den untersuchten Anwendungsfällen entweder realisiert werden, oder zum Verständnis benötigt werden. Einige dieser Sicherheitskriterien sind insbesondere von Notwendigkeit für die Klassifizierung. In Kapitel 5 werden die untersuchten Studien vorgestellt, allerdings nicht in ihrem vollem Umfang, sondern in Bezug auf den Einsatz homomorpher Kryptosysteme, der Gründe für den Einsatz und dem Umgang mit der Verformbarkeit von Chiffretexten durch einen Angreifer.

2 Grundlegende Aspekte

In diesem Kapitel werden die Grundlagen eingeführt um das Schema eines semihomomorphen Kryptosystems zu verstehen. Dazu wird zunächst der Zweck eines Kryptosystems erläutert und der Begriff formal eingeführt. Dann wird die Definition erweitert bis hin zum semihomomorphen Kryptosystem.

2.1 Schutzziele der Informationssicherheit

Die Kryptografie hat zur Aufgabe Lösungen für die Realisierung verschiedener Schutzziele der Informationssicherheit bei der Speicherung, Vervielfältigung und Übertragung von Informationen umzusetzen. Die Kryptografie stellt dazu verschiedene Algorithmen und Protokolle bereit. Grundlegende Schutzziele beim Übertragen von Informationen zwischen mehreren Parteien in Nachrichten sind nach [MVOV96, p.4][DKK02, p.2]:

1. **Vertraulichkeit:** Keine unautorisierte Kenntnisnahme. Nur dazu berechtigte Personen sollen eine bestimmte Information lesen können oder Zugang zu dieser Information erhalten. Dieses Sicherheitsziel impliziert Geheimhaltung. Vertraulichkeit kann physisch erreicht werden (z.B. das Abschließen in einer Kiste) oder durch mathematische Algorithmen, welche die Daten unverständlich machen.
2. **Integrität:** Keine unautorisierte unbemerkte Datenmanipulation. Um die Integrität von Daten zu gewährleisten muss die Möglichkeit einer Detektion von Veränderungen in den Daten realisiert werden. So kann insbesondere das Hinterlegen von Falschdaten in einer Nachricht oder das Fehlen von Teilen einer Nachricht erkannt werden.
3. **Authentizität:** Authentizität meint die Fähigkeit einer Identifikation in Bezug auf die Information als auch auf die Kommunikationspartner (Entitäten). Fordert man, dass die kommunizierenden Teilnehmer in der Lage sein sollen, sich gegenseitig zu identifizieren, spricht man von Authentizität der Entitäten. Bei einseitiger Kommunikation fordert man lediglich, den Urheber einer Nachricht identifizieren zu können, also die Authentizität des Datenursprungs.
4. **Nichtabstreitbarkeit:** Der Versand einer Nachricht kann von dem Sender nach dem Versand nicht mehr abgestritten werden. Zum Beispiel, wenn eine Entität den Kauf in einer unabstreitbaren Nachricht zunächst autorisiert, jedoch später verneint, so kann in Konfliktfällen die ursprüngliche Zusage nachgewiesen werden.

2.2 Kryptosysteme

Ein Kryptosystem besteht aus mehreren Algorithmen, um das Schutzziel der Vertraulichkeit bei der Übertragung von Informationen umzusetzen [Cry].

Damit ermöglicht ein Kryptosystem zwei Parteien Alice und Bob über einen ungeschützten Kanal, in dem die Nachricht übertragen wird, zu kommunizieren ohne das eine dritte Partei Zugang zu der geschützten Information erhält.

Die zugrundelegenden Algorithmen und resultierende Eigenschaften über die Beziehung von Klartexten zu Chiffretexten führen zu verschiedenen Klassen von Kryptosystemen. Einige dieser Kryptosysteme führen wir in diesem Abschnitt ein.

In der Literatur lassen sich verschiedene Formalisierungen für ein Kryptosystem finden. Wir verwenden die Definition von Douglas R. Stinson [Stio6, p.1] und erweitern diese Definition im Anschluss für eine bessere Differenzierung um Eigenschaften wie Determinismus oder Symmetrie.

Definition 1 (Kryptosystem)**.** *Ein Kryptosystem ist ein Quintupel* $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ *welches folgenden Eigenschaften genügt:*

1. $\mathcal{P}$ *ist eine endliche Menge von Klartexten, der Klartextraum.*
2. $\mathcal{C}$ *ist eine endliche Menge von Chiffretexten, der Chiffreraum.*
3. $\mathcal{K}$ *ist eine endliche Menge möglicher Schlüssel in Form von Tupeln, der Schlüsselraum.*
4. *Für alle Schlüssel* $k \in \mathcal{K}$ *gibt es eine Verschlüsselungsfunktion* $\mathcal{E} \ni e_k : \mathcal{P} \to \mathcal{C}$ *und zugehörige Entschlüsselungsfunktion* $\mathcal{D} \ni d_k : \mathcal{C} \to \mathcal{D}$*, so dass für alle Klartexte* $x \in \mathcal{P}$ *folgende Identität gilt:* $d_k(e_k(x)) = x$*. Dabei können in der Verschlüsselungs- und Entschlüsselungsfunktion nur ein Teil des Tupels k verwendet werden.*

Grundsätzlich muss ein Kryptosystem also mindestens drei Algorithmen bereitstellen: Einen für die Erzeugung des Schlüssels, einen für die Verschlüsselung und einen für die Entschlüsselung. Man beachte, dass der Schlüssel k als *ein* Element des Schlüsselraums selber aus *mehreren* Elementen zusammengesetzt werden kann. Diese Eigenschaft führt zu den nächsten beiden Erweiterungen. Wir definieren:

Definition 2 (Symmetrisches Kryptosystem)**.** *Sei* $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ *ein Kryptosystem. Dann nennen wir K symmetrisch, wenn jede Verschlüsselungsfunktion* e_k *als auch die zugehörige Entschlüsselungsfunktion* d_k *vollständig von demselben Schlüssel* $k \in \mathcal{K}$ *abhängen. Vollständig bedeutet, dass diese Funktionen insbesondere nicht nur von einer Teilmenge von k abhängen, wenn der Schlüssel k aus mehreren Parametern zusammengesetzt ist. Alle Parameter von k gehen in die Verschlüsselungsfunktion und Entschlüsselungsfunktion ein.*

Ein Nachteil von symmetrischen Kryptosystemen liegt auf der Hand: Da sowohl Alice als auch Bob den gleichen geheimen Schlüssel benötigen, muss dieser über einen sicheren Kanal übertragen werden bevor sie geheime Nachrichten austauschen können. Daher sind symmetrische Kryptosysteme auch bekannt als *private-key* Kryptosysteme.

Im Gegensatz dazu existieren Kryptosysteme bei denen k aus einem privaten und einem öffentlichen Teilschlüssel zusammensetzt ist. Diese Teilmengen müssen nicht disjunkt[1] sein. Alice kann nun ihren öffentlichen Teilschlüssel bekannt geben, um so Dritten zu ermöglichen ihr Informationen vertraulich zukommen zu lassen. Daher spricht man auch von *public-key* Kryptosystemen, ein Konzept das Ursprünglich von Diffie und Hellmann in [DH76, p.648] eingeführt wurde. Wir definieren:

Definition 3 (Asymmetrisches Kryptosystem)**.** *Sei* $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ *ein Kryptosystem. Dann nennen wir K asymmetrisch wenn sich der Schlüssel* $k \in \mathcal{K}$ *zusammensetzt aus* $k = (k_s,k_p)$ *mit* $k_s,k_p \in K$*. Die*

[1] Im RSA-Kryptosystem enthalten die Mengen beider Teilschlüssel den Modulus n [RSA78, p.6].

Verschlüsselungsfunktion ist dann $\mathcal{E} \ni e_{k_p} : \mathcal{P} \to \mathcal{C}$, während die Entschlüsselungsfunktion $\mathcal{D} \ni d_{k_s} : \mathcal{P} \to \mathcal{C}$ ist. Während e_k von beliebigen Parteien ausgeführt werden kann, kann d_k nur vom Besitzer des privaten Teilschlüssel k_s ausgeführt werden. k_s muss geheim gehalten werden.

Öffentlicher und privater Teilschlüssel stehen in einem mathematischen Zusammenhang, der jedoch für Angreifer mit begrenzter Rechenkapazität praktisch unmöglich zu erschließen ist.

Diese drei Definitionen genügen noch nicht um zu beschreiben, in welcher Beziehung Klartexte zu ihren Chiffraten stehen, wenn sie mit dem gleichen Schlüssel in verschiedenen Instanziierungen des Verschlüsselungsalgorithmus erzeugt werden. Dies ist für mögliche Angreifermodelle der Kryptoanalyse von Bedeutung welche in 3.1 vorgestellt werden.

Definition 4 (Deterministisches Kryptosystem)**.** *Sei $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ ein Kryptosystem. Dann nennen wir K deterministisch, wenn gilt: Für einen beliebigen festen Schlüssel $k \in \mathcal{K}$ ist e_k injektiv.*

Seien nun $c_1,c_2 \in \mathcal{C}, c_1 = c_2$ zwei Chiffrate unter e_k, dann folgt daraus für ihre Klartexte, dass $x_1 = x_2$. Also führt der gleiche Klartext unter Verwendung desselben Schlüssels bei verschiedenen Ausführungen von der Verschlüsselungsfunktion e_k zu einem identischen Chiffrat.

Jetzt kann man in Abgrenzung zu dieser Definition das probabilistische Kryptosystem einführen. Ein probabilistisches Kryptosystem erzeugt für gleiche Klartexte bei demselben Schlüssel mit jeder Ausführung der Verschlüsselungsfunktion ein anderes Chiffrat.

Das Konzept eines probabilistischen Kryptosystems wurde ursprünglich von Goldwasser und Micali eingeführt in [GM82]. Wir definieren in Anlehnung an [Sti06, p.345]:

Definition 5 (Probabilistisches Kryptosystem)**.** *Ein Probabilistischen Kryptosystem ist ein sechselementiges Tupel $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D},\mathcal{R})$. Wie bereits in Definition 1 ist $\mathcal{P}$ ist der Klartextraum, $\mathcal{C}$ der Chiffreraum und $\mathcal{K}$ der Schlüsselraum. Neu sind:*

- *$\mathcal{R}$ ist eine endliche Menge von Zufallszahlen*
- *Für alle Schlüssel $k \in \mathcal{K}$ gibt es eine Verschlüsselungsfunktion $\mathcal{E} \ni e_k : \mathcal{P} \times \mathcal{R} \to \mathcal{C}$ und zugehörige Entschlüsselungsfunktion $\mathcal{D} \ni d_k : \mathcal{C} \times \mathcal{R} \to \mathcal{D}$, so dass für alle Klartexte $x \in \mathcal{P}$ und alle Zufallszahlen $r \in \mathcal{R}$ folgende Identität gilt: $d_k(e_k(x,r)) = x$*

Für ein festes $k \in K$ und ein beliebigen Klartext $x \in P$ definieren wir die Wahrscheinlichkeitsverteilung $Pr_{K,x}$ auf C, so dass $Pr_{K,x}(y)$ die Wahrscheinlichkeit angibt, dass y ein Chiffrat von x unter e_k ist. Nun fordern wir:

- *Gegeben $x_1,x_2 \in P, x_1 \neq x_2$ und $k \in K$. Dann sind die Wahrscheinlichkeitsverteilungen Pr_{K,x_1} und Pr_{K,x_2} nicht in Polynomialzeit unterscheidbar[2].*

Nicht unterscheidbar hat insbesondere zur Folge, dass wir auch nicht wissen ob die gleiche Wahrscheinlichkeitsverteilung hinter zwei Chiffraten steckt. In anderen Worten: Die wiederholte Verschlüsselung eines Klartextes führt im Allgemeinen zu verschiedenen Chiffraten.

Ein probabilistisches Kryptosystem nutzt Zufall in der Verschlüsselungsfunktion, so dass der gleiche Klartext verschieden verschlüsselt wird. Probabilistischen Kryptosystemen sind meistens gleichzeitig asymmetrisch, es ist jedoch auch möglich mit symmetrischen Verschlüsselungsverfahren diese

[2] unterscheidbar meint, dass ein Algorithmus existiert der entscheiden kann ob ein Bitstring eher aus der Wahrscheinlichkeitsverteilung Pr_{K,x_1} oder Pr_{K,x_2} kommt [Sti06, p.329]. Die Berechenkomplexität des Algorithmus wächst in Relation zur Eingabegröße höchstens in Größe einer Polynomfunktion.

Eigenschaft zu erreichen, z.B. bei Verwendung von Blockchiffren im Cipher Block Chaining Mode. Die Menge der Zufallszahlen $\mathcal{R}$ entspricht dann der Menge möglicher Initialisierungsvektoren.

2.3 Algebraische Strukturen

Nun wird der Begriff eines homomorphen Kryptosystems im Detail eingeführt. Um dessen Anwendung zu verstehen, ist es jedoch zunächst nötig folgende algebraische Strukturen nach [Fis10, p.43,54,56] zu erläutern, um nachvollziehen zu können welche mathematischen Rechenoperationen ein homomorphes Kryptosystem bietet.

2.3.1 Von Gruppen zu Körpern

Definition 6 (Gruppe). *Eine Gruppe ist ein Tupel $(G,+)$ bestehend aus der Menge G und einer Verknüpfung $+$ auf G mit folgenden Eigenschaften:*

- *$+$ ist assoziativ $\forall a,b,c \in G : (a+b)+c = a+(b+c)$*
- *Es existiert bzgl. $+$ ein neutrales Element e_+ in G: $\forall a \in G : e_+ + a = a$*
- *Jedes g in G ist invertierbar mit einem Element g' aus G, s.d. $g+g' = e_+$*

Ist die Verknüpfung einer Gruppe zusätzlich kommutativ ($\forall a,b \in G : a+b = b+a$), so nennt man sie abelsch.

Definition 7 (Ring). *Ein Ring ist ein Tripel $(R,+,\cdot)$ bestehend aus der Menge R und zwei Verknüpfungen $+$ und $\cdot$ auf R mit folgenden Eigenschaften:*

- *$(R,+)$ eine abelsche Gruppe.*
- *$\cdot$ ist assoziativ für alle Elemente aus R*
- *Es gelten die Distributivgesetze: $\forall a,b,c \in R : a\cdot(b+c) = (a\cdot b)+(a\cdot c)$ und $(a+b)\cdot c = (a\cdot c)+(b\cdot c)$*

Hat R bzgl. $\cdot$ ein neutrales Element $e_\bullet$, so nennen wir R einen „Ring mit Eins“ (oder Monoid). Ist R bzgl. $\cdot$ kommutativ, so nennen wir R einen „kommutativen Ring“.

Definition 8 (Körper). *Sei K ein kommutativer Ring mit Eins wie in 7, so heißt K Körper, wenn $(R\backslash\{e_+\},\cdot)$ ein abelsche Gruppe bildet.*

2.3.2 Homomorphismus

Nun haben wir gesehen wie Elemente einer Menge mit bestimmten Rechenoperationen und mathematischen Eigenschaften in algebraische Strukturen zusammengefasst werden können. Werden Elemente einer algebraischen Struktur unter einer Funktion in einer andere Struktur abgebildet, so können für die Elemente unter der Abbildung diese Eigenschaften erhalten bleiben. Ist das der Fall, so spricht man von einem Homomorphismus. Homomorphismen existieren für alle oben genannten Strukturen. Einen Ringhomomorphismus definiert man formal zu [Fis10, p.56]:

Definition 9 (Ringhomomorphismus). *Sind $(R,+_1,\bullet_2)$ und $(S,+_2,\bullet_2)$ Ringe, so nennt man die Abbildung $\phi : R \to S$ einen Ringhomomorphismus, falls gilt:*

- *$\forall a,b \in R : \phi(a +_1 b) = \phi(a) +_2 \phi(b)$ und $\phi(a \bullet_2 b) = \phi(a) \bullet_2 \phi(b)$*

2.3.3 Modulare Arithmetik

In der Kryptographie rechnet man für gewöhnlich nicht mit allen Zahlen, sondern mit *natürlichen Zahlen unterhalb einer gewissen Grenze n*. Rechenergebnisse mit Zahlen größer als n werden auf Zahlen bis zur Grenze n *modular reduziert*. Das Rechnen auf dem reduzierten Zahlenbereich nennt man modulare Arithmetik.

Von besonderer Bedeutung sind insbesondere Prime Restklassen $(\mathbb{Z}/n\mathbb{Z})^*$. Diese Struktur ist eine Menge bestehend aus ganzen Zahlen kleiner n, welche gleichzeitig zu n teilerfremd sind:

$$\mathbb{Z}/n\mathbb{Z} = \left\{a \in \{0,1,\ldots,n-1\} |\ ggT(a,n) = 1\right\}$$

Zusammen mit der Multiplikation modulo n bildet die Prime Restklasse eine Gruppe, d.h. ihre Elemente sind insbesondere invertierbar [BSW95, p.116].

2.4 Homomorphe Kryptosysteme

Mit den mathematischen Voraussetzungen aus 2.3 können wird nun den Begriff eines homomorphen Kryptosystems formalisieren.

2.4.1 Semihomomorphes Kryptosystem

Laut 2.3.2 ist ein Homomorphismus strukturerhaltend, das bedeutet: Für Verknüpfungen im Chiffreraum findet man eine entsprechende Verknüpfung im Klartextraum. Diese Eigenschaft in einem Kryptosystem ermöglicht eine Verarbeitung von Daten unter Wahrung der Vertraulichkeit dieser. Daher können semihomomorphe Kryptosysteme eingesetzt werden, um bekannte Protokolle datenschutzfreundlich zu machen, wie mehrere der vorgestellten Veröffentlichungen in Kapitel 5 zeigen. Auf der anderen Seite kann die Homomorphieeigenschaft bösartig ausgenutzt werden, wie in 3.2.4 an einem Beispiel erläutert wird.

Definition 10 (Semihomomorphes Kryptosystem)**.** *Sei $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ ein asymmetrisches Kryptosystem. Wir nennen K semihomomorph, wenn $(\mathcal{P},\oplus)$ und $(\mathcal{C},\odot)$ Gruppen bilden und die Verschlüsselungsfunktion ein Gruppenhomorphismus ist.*

- *Das heißt alle unter e_{k_p} erzeugten Chiffrate bilden einen Gruppe.*
- *Sei $e_{k_p}(x_1) = c_1, e_{k_p}(x_2) = c_2$. Dann gilt: $d_{k_s}(c_1 \odot c_2) = m_1 \oplus m_2$*

Diese Definition ist orientiert an [KL14, p.499]. Ein konkretes Beispiel stellen wir mit dem Okamoto-Uchiyama Schema in 2.4.4 vor.

2.4.2 Vollhomomorphes Kryptosystem

Lange Zeit nahm man an, dass es keine Kryptosysteme gibt, welche beliebige Sequenzen von Rechenoperationen im Chiffreraum ermöglichen - d.h. vollhomomorph wären. Bekannte semihomomorphe Kryptosysteme erlauben nur eingeschränkte Operationen (z.B. Addition oder XOR). Dann stellten Boneh et al. ein Verfahren vor, welches neben beliebig vielen Additionen zusätzlich *eine* Multiplikation erlaubt [BGN05, p.2]. Im Jahr 2009 stellte schließlich Craig Gentry [Gen09] erstmals ein Verfahren vor, wo im Chiffreraum beliebig viele Rechenoperationen eines Rings möglich sind, d.h. Additionen und Multiplikationen. In Anlehnung an [YPB14, p.48] wird definiert:

Definition 11 (Vollhomomorphes Kryptosystem). *Sei $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ ein asymmetrisches Kryptosystem. Wir nennen K vollhomomorph, wenn $(P,\otimes,\ominus)$ und $(C,\odot,\oplus)$ Ringe bilden und die Verschlüsselungsfunktion ein Homomorphismus von Ringen ist.*

- *Das heißt alle unter e_{k_p} erzeugten Chiffrate bilden einen Ring.*
- *Sei $e_{k_p}(x_1) = c_1, e_{k_p}(x_2) = c_2$. Dann gilt: $d_{k_s}(c_1 \odot c_2) = m_1 \otimes m_2$ sowie $d_{k_s}(c_1 \oplus c_2) = m_1 \ominus m_2$*

Gentry konnte zeigen, dass ein beliebe Schaltung aus NANDs im Chiffreraum evaluiert werden kann. Dies ist ein Durchbruch, denn NANDs bilden für sich ein vollständiges Operatorensystem mit dem jede boolesche Funktion beschrieben werden kann [Hof10, p.129]. Damit ermöglicht ein vollhomomorphes Kryptosystem die Berechnung einer *beliebigen* booleschen Funktion im Chiffreraum.

Ein vollhomomorphes Kryptosystem besteht aus vier Algorithmen: Zusätzlich zur Schlüsselgenerierung, Verschlüsselung und Entschlüsselung gibt es noch einen Algorithmus für die Evaluierung eines Schaltkreises im Chiffreraum. Sei k_p der öffentliche Schlüssel unter dem die Chiffrate $c_1,\ldots,c_l$ erzeugt worden. Dann ermöglicht der Evaluierungsalgorithmus das Berechnen eines beliebigen Schaltkreises S zu dem Ergebnis c:

$$\textbf{Evaluate}_{k_p}(S, c_1,\ldots,c_l) = c$$

Trotz dieser größeren Funktionalität finden bis heute vollhomomorphe Kryptosysteme wenig Einsatz in der Praxis, da sie zu hohe Anforderungen an verfügbare Rechenkapazitäten stellen. So führen z.B. Sicherheitsgarantien gegen bekannte Angriffe bei Gentrys Verfahren dazu, dass die Expansionsrate[3] der Chiffretext überhand gewinnt [YPB14, p.49].

Stattdessen weicht man auf semihomomorphe Kryptosysteme aus wenn sie den Anforderungen genügen, oder realisiert eine Kombination aus zwei Kryptosystemen, von denen eins additiv und das andere multiplikativ ist. Ein Beispiel für so eine Realisierung werden wir in 5.1 sehen. Eine andere Möglichkeit ist der Einsatz von eingeschränkt vollhomomorphen Kryptosystemen die im nächsten Abschnitt vorgestellt werden.

Abschließende Anmerkungen:
Es kommen fast ausschließlich probabilistische homomorphe Kryptosysteme zum Einsatz, obwohl es deterministische homomorphe Kryptosysteme gibt. Jedoch wurde von Boneh und Lipton in 1996 gezeigt, das *jedes* deterministische homomorphe Kryptosystem in subexponentieller Zeit gebrochen werden kann [BL96].

2.4.3 Eingeschränkt Vollhomomorphes Kryptosysteme

Bei eingeschränkt vollhomomorphen Kryptosystemen (in der englischen Literatur werden die Begriffe *leveled* oder *somewhat homomorphic encryption* verwendet) geht man einen Kompromiss ein, um den Overhead von vollhomomorphen Kryptosystemen zu reduzieren, unter Verzicht nicht mehr beliebig viele Rechenoperationen durchführen zu können. Übertragen auf Gentrys Evaluierung eines Schaltkreises, wird die Tiefe der Hintereinanderschaltungen von Schaltkreiselementen eingeschränkt.

[3] Unter der Expansionsrate versteht man das Verhältnis von der Länge des Chiffrats zu der Länge des zugehörigen Klartextes. Die Expansionsrate von probabilistischen Kryptosystemen ist sehr groß, d.h. ein Kilobit wird benötigt und einige Bits zu verschlüsseln. [NS98, p.3+10]

Für eine genauere Differenzierung von *leveled* zu *somewhat* vollhomomorphen Kryptosystemen, sowie eine Übersicht zu verwandten Begrifflichkeiten im Zusammenhang mit vollhomomorphen Kryptosystemen sei der Leser auf [ABC+15, p.5+14] verwiesen.

2.4.4 Okamoto-Uchiyama Kryptosystem

In Kapitel 5 werden verschiedene homomorphe Kryptosysteme erwähnt, welche teilweise verwandt sind, wie die Tabelle in 5.8 zeigt. Einige dieser Verfahren können als Generalisierungen anderer aufgefasst werden. Im Folgenden wird nun beispielhaft das homomorphe Kryptosystem von Okamoto-Uchiyama [OU, p.311] vorgestellt.

Das Kryptosystem von Okamoto-Uchiyama ist probabilistisch, asymmetrisch, verfügt über Homomorphieeigenschaften und ist semantisch sicher (3.2.2). Das System arbeitet in der Prime Restklasse $(\mathbb{Z}/n\mathbb{Z})^*$ (2.3.3) wobei n sich zusammensetzt aus zwei großen Primzahlen p,q über $n = p^2q$. Wie in 2.2 erwähnt, kommen in einem Kryptosystem drei grundlegende Algorithmen für Schlüsselgenerierung, Verschlüsselung und Entschlüsselung vor:

Schlüsselgenerierung:

1. Wähle zwei große Primzahlen p,q $(|p| = |q| = k)$.
2. Dann setze $n = p^2q$.
3. Wähle $g \in (\mathbb{Z}/n\mathbb{Z})^*$, so dass für die Ordnung[4] g_p von g gilt: $g_p := g^{p-1} \bmod p^2 \stackrel{!}{=} p$
4. Dann setze $h = g^n \bmod n$.

Öffentlicher Schlüssel ist das Tupel (n,g,h), privater Schlüssel ist das Tupel (p,q).

Verschlüsselung:

Sei $0 < m < 2^{k-1}$ ein Klartext.

1. Wähle gleichverteilt ein $r \in (\mathbb{Z}/n\mathbb{Z})$. Diese Ergänzung macht das Kryptosystem probabilistisch. Bei nochmaliger Verschlüsselung bekommt der gleiche Klartext ein anderes Chiffrat.
2. Dann ist das zu m zugehörige Chiffrat:

$$E(m,r) = g^m h^r \bmod n = C.$$

Entschlüsselung:

Sei C_p die Ordnung von dem Chiffrat C modulo p^2. Dann ist der zu C zugehörige Klartext:

$$D(C) = \frac{C_p - 1}{g_p - 1} \bmod p = m.$$

[4] Dies ist die kleinste natürliche Zahl, für die ein Element der Gruppe mit sich selbst multipliziert das neutrale Element der Gruppe ergibt.

Eigenschaften des Okamoto-Uchiyama Kryptosystems

Homomorphe Addition von Klartexten: Das Verfahren von Okamoto-Uchiyama ermöglicht eine homomorphe Addition von Klartexten m_1, m_2 durch die Multiplikation ihrer jeweiligen Chiffren unter der Nebenbedingung, dass $m_1 + m_2 < p$.

$$E(m_1, r_1)E(m_2, r_2) \bmod n = g^{m_1} h^{r_1} g^{m_2} h^{r_2} \bmod n = g^{m_1+m_2} h^{r_3} \bmod n = E(m_1 + m_2, r_3) \bmod n$$

Die Addition kann auf mehrere Klartexte $m_1, m_2, \ldots, m_n$ erweitert werden, sofern ihre Summe weiter die Nebenbedingung $\sum_{i=1}^{n} m_i < p$ erfüllt.

Homomorphe Multiplikation von Klartexten: Weiter ist eine homomorphe Multiplikation von einem beliebigen zu verschlüsselndem Klartext m mit einer Konstanten k möglich, indem das Chiffrat zu m mit k potenziert wird:

$$E(m, r)^k = (g^m h^r)^k \bmod n = g^{mk} h^{r'} \bmod n.$$

Ersetzbarkeit von Chiffraten: Ein erzeugtes Chiffrat $C = E(m, r)$ kann alleine durch Kenntnis des öffentlichen Schlüssels in ein von C verschiedenes Chiffrat C' gewandelt werden, so dass die Beziehung zueinander verborgen, jedoch ihre Klartexte äquivalent sind:

$$C' = Ch^{r'} \bmod n = (g^m h^r)h^{r'} \bmod n = g^m h^{r''} \bmod n = E(m, r'').$$

3 Sicherheitseigenschaften

In diesem Kapitel werden Begriffe eingeführt, welche die Sicherheit kryptographischer Verfahren formalisieren. Diese Sicherheitseigenschaften sind Kriterien, anhand derer ein Kryptosystem bewertet wird.

Die Sicherheitsbegriffe für Kryptosysteme, welche jetzt vorgestellt werden, lassen sich grundsätzlich wie folgt einteilen:

- Sicherheitsbegriffe, die *Angreifermodelle* definieren
- Sicherheitsbegriffe, die ein wünschenswertes *Sicherheitsziel* definieren

Diese Einteilung wurde ursprünglich von Moni Noar vorgeschlagen[1].

Betrachtet man ein asymmetrisches Kryptosystem, bei dem der öffentliche Schlüssel bekannt ist, so ermöglicht dies einem Angreifer jeden beliebigen Klartext unter diesem Schlüssel zu verschlüsseln. Diese Fähigkeit des Angreifers führt zu dem Angreifermodell *Adaptiver Angriff mit ausgewähltem Klartext* in Abschnitt 3.1. Ein zu erreichendes Sicherheitsziel bei einem Kryptosystem ist die Vermeidung einer *Verformung von Chiffretexten durch unautorisierte Dritte*, welches in Abschnitt 3.2.4 definiert wird.

3.1 Angreifermodelle der Kryptoanalyse

Angriffe auf ein Kryptosystem versuchen die vertrauliche Information, den verschlüsselten Klartext, aus einem Chiffretext zu rekonstruieren oder den eingesetzten Schlüssel zu rekonstruieren.

Bei der Beurteilung der Sicherheit von kryptographischen Verfahren geht man in der modernen Kryptoanalyse davon aus, dass der Angreifer die Algorithmen kennt, welche ein Kryptosystem einsetzt. Dieser Gedanke wurde von Shannon erstmals formuliert [Sha49, p.662]. Damit soll die Sicherheit eines Kryptosystems nur von dem Schlüssel abhängen, was auch als das Kerckhoffsche Prinzip bekannt ist.

Im Folgenden werden verschiedene Angreifermodelle der Kryptoanalyse vorgestellt. Diese unterscheiden sich dadurch, auf welche Informationen der Angreifer zurückgreifen kann, um Informationen aus Chiffraten zu gewinnen. Dabei betrachten wir diejenigen Angreifermodelle, welche für asymmetrische Verfahren relevant sind.

Die Angreifermodelle werden in der Reihenfolge nach der Stärke des Angreifers vorgestellt, angefangen mit dem schwächsten [DKK02, p.4-5][BSW95, p.7-8]:

[1] Dies wird in [BDPR] erwähnt und hier wird eine Quelle zitiert welche nicht ausfindig gemacht werden konnte: M. Noar, private communication, März 1998. Diese Aufteilung findet sich jedoch auch so in [Sma03, p.289]: Notions of security, Notions of attacks.

Reiner Chiffretext-Angriff (*ciphertext-only attack*/COA): Bei dieser Angriffsform ist der Angreifer in der Lage Zugriff auf Chiffretexte einer vertraulichen Kommunikation zu bekommen. Den Chiffretext kann sich der Angreifer nicht auswählen, da er nicht in die Kommunikation eingreift. Bei einem reinen Chiffretext-Angriff handelt es sich um die schwächste Form eines Angreifers. Kann ein Angreifer ohne Aufwand unter diesem Angreifermodell Einsicht in die verborgenen Klartexte gelangen, so bietet das Kryptosystem keinen Schutz - es ist unsicher.

Angreifer mit bekannten Klartext (*known-plaintext attack*/KPA): Der Angreifer hat Zugriff auf eine begrenzte Zahl aus Chiffretexten und zugehörigen Klartexten. Anhand der Beziehungen welche zwischen den Klartext-Chiffretext-Paaren besteht, versucht der Angreifer Kenntnisse über die Algorithmen des Kryptosystems zu gewinnen. Mit den gewonnenen Erkenntnissen versucht er dann Chiffretexte zu entschlüsseln, für welche kein passender Klartext vorliegt. Dieser Angriff ist möglich in Situationen, in welchen im Klartext mit standardisierten Formaten kommuniziert wird. So beginnen z.B. PDF-Dateien im Header mit einem fest definierten String, z.B. „%PDF-1.7“. Dieser gibt die Version des Portable Document Formats an [Ado06, p.92]. Im zweiten Weltkrieg wurde Enigma mit bekannten Klartexten angegriffen [Sch11].

Angreifer mit ausgewählten Klartext (*chosen-plaintext attack*/CPA1): Der Angreifer ist in der Lage *einmal* Chiffretexte zu einer Menge ausgewählter Klartexte zu erzeugen. Information die daraus gewonnen werden kann, wird benutzt um unbekannte Chiffretexte zu entschlüsseln. Es ist dem Angreifer insbesondere nicht möglich, die Ergebnisse der Analyse eines Klartext-Chiffretext-Paars zu benutzten um ein weiteres Paar zu erzeugen. Alle Paare erhält der Angreifer am Anfang des Angriffs zusammen. Im Falle von asymmetrischen homomorphen Kryptosystemen muss dieses Angreifermodell grundsätzlich betrachtet werden, da der öffentliche Schlüssel frei verfügbar ist. Damit hat ein Angreifer eines asymmetrischen homomorphen Kryptosystems *immer* die Möglichkeit ausgewählte Klartexte anzugreifen.

Adaptiver Angreifer mit ausgewählten Klartext (*adaptive chosen-plaintext attack*/CPA2): Dieses Angreifermodell ist eine Erweiterung des Angriffs mit ausgewähltem Klartext. Dabei kann der Angreifer nach Erhalt eines Klartext-Chiffretext-Paares ein weiteres erzeugen und dabei Information aus der Analyse des vorherigen Klartext-Chiffretext-Paares ausnutzen. Damit ist dieser Angriff im Vergleich zu CPA1 *interaktiv*. Der Angreifer hat Zugriff zu einem Verschlüsselungsapparat[2] oder kann diesen mehrmals verwenden.

Angreifer mit ausgewählten Chiffretext (*chosen-ciphertext attack*/CCA1): Analog zu CPA1 definiert man: Der Angreifer hat *einmal* Zugriff zu Klartexten von ausgewählten Chiffretexten durch einen Entschlüsselungsapparat. Bleichenbacher präsentiert in [Ble] einen Angriff mit ausgewählten Chiffretexten, der erfolgreich ein RSA-basiertes Protokoll aus dem SSL 3.0 Standard bricht.

Adaptiver Angreifer mit ausgewählten Chiffretext (*adaptive chosen-ciphertext attack*/CCA2): Wie bei dem Angreifermodell in CPA2 ist der Angriff interaktiv. Der Angreifer kann Klartexte zu ausgewählten Chiffretexten erhalten und anschließend erneut mit dem Entschlüsselungsapparat einen weiteren Chiffretext entschlüsseln. Bei dem Schutz vor adaptiven Angriffen mit ausgewählten Chiffretexten handelt es sich um die stärkste mögliche Anforderung an ein Kryptosystem.

[2] Verschlüsselungs- und Entschlüsselungsapparate sind Funktionen die dem Angreifer ermöglichen Klartexte zu verschlüsseln und Chiffretexte zu entschlüsseln, in deren Funktionsweise der Angreifer jedoch keine Einsicht hat. Daher spricht in der Literatur auch von einer Black Box [Sma03, p.291] oder einem Entschlüsselungsorakel [BDPR, p.2].

3.1.1 Anmerkungen zu den Angreifermodellen

In der Literatur werden diese Angriffe in passive und aktive Angreifermodelle unterteilt. Jedoch führen die Definitionen zu verschiedenen Aufteilungen: Buchmann unterscheidet Angreifer danach, ob sie nur Chiffretexte (passiv) oder gleichzeitig ausgewählte Klartexte und Chiffretexte (aktiv) angreifen können [Buco4, p.76]. Smart dagegen bezeichnet in [Smao3, p.291] den Angriff mit ausgewähltem Klartext als passiv.

Statt der Angreifermodelle CPA1 und CPA2 ist der Gebrauch der Bezeichnung CPA üblich [AKP13]. Damit ist CPA2-Sicherheit gemeint, da die Verschlüsselungsfunktionen der eingesetzten Kryptosysteme bekannt sind. Die Kryptosysteme, welche in dieser Bachelorarbeit behandelt werden, sind öffentlich bekannt. Daher wird im weiteren Verlauf nur von CPA-Sicherheit gesprochen, um die Angreifermodelle CPA1 und CPA2 gleichermaßen abzudecken.

3.2 Sicherheitszeile

Im Folgenden wird formalisiert, inwiefern ein Angreifer Einsicht in Chiffretexte erhält durch Angabe seines Vorteils (engl. *advantage*) gegenüber einem ratenden Angreifer.

3.2.1 Perfekte Sicherheit

In einem Kryptosystem mit perfekter Sicherheit (engl. *perfect security*) ist es für einen Angreifer mit unendlich großer Rechenkapazität nicht möglich, etwas über den Klartext anhand eines vorliegenden Klartextes zu lernen.

Definition 12 (Perfekt Sicheres Kryptosystem)**.** *Sei $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ ein Kryptosystem. Dann ist K perfekt sicher, wenn für alle $x \in \mathcal{P}$ und alle $c \in \mathcal{C}$ gilt:*

$$Pr(P = x \mid C = c) = Pr(P = x).$$

[Smao3, p.280]

Laut dem Theorem von Shannon ist dies genau dann der Fall, wenn 1) alle Schlüssel $k \in \mathcal{K}$ des Schlüsselraums gleichverteilt gezogen werden und 2) jeder Schlüssel nur einmal verwendet wird. [KL14, p.36] Aus diesem Grund sind asymmetrische Kryptosysteme im praktischen Einsatz nicht perfekt sicher, da man den gleichen Schlüssel für mehrere Klartexte verwendet.

3.2.2 Semantische Sicherheit

Der Begriff semantischer Sicherheit (engl. *semantic security*) unterscheidet sich von perfekter Sicherheit darin, dass die Rechenkapazitäten des Angreifers nach oben polynomiell beschränkt sind [Smao3, p.290] [Buco4, p.166].

Auf eine formale Definition wird in diesem Abschnitt verzichtet, denn semantische Sicherheit ist gleichbedeutend zu der Unterscheidbarkeit von Chiffretexten [Gol93, p.8][DKK02, p.203]. Diese wird im nächsten Abschnitt eingeführt.

3.2.3 Ununterscheidbarkeit von Chiffretexten (ciphertext indistinguishability/IND)

Gegeben zwei Klartexte x_1, x_2 und ein Chiffretext c_b von einem der beiden Klartexte bedeutet Unterscheidbarkeit von Chiffretexten (engl. *ciphertext indistinguishability*, kurz: IND), dass es einem Angreifer $\mathcal{A}$ nicht möglich ist c_b dem zugehörigen Klartext zuzuordnen. Damit meint man, dass der Angreifer diese Zuordnung mit einer Wahrscheinlichkeit treffen muss, die größer ist, als wenn er zufällig raten würde (hier: $> \frac{1}{2}$). Es wird dem Angreifer jedoch erlaubt, die Länge des Klartextes anhand des Chiffretextes zu lernen. Daher wird im Folgenden vorausgesetzt, dass die Klartexte x_1, x_2 immer die gleiche Länge haben [BDPR, p.32].

Wäre das verwendete Kryptosystem deterministisch, so würde dem Angreifer diese Zuordnung durch Evaluierung der Verschlüsselungsfunktion gelingen. Mit dem öffentlichen Schlüssel k_p berechnet der Angreifer $c_i = e_{k_p}(x_i)$. Durch einen Vergleich mit dem erhaltenen Chiffretext c_b kann der Angreifer b bestimmen. Daher muss bei einem IND-sicherem Kryptosystemen die Verschlüsselungsfunktion probabilistisch sein, denn sonst gelingt dem Angreifer diese Zuordnung.

Formalisiert wird die Ununterscheidbarkeit von Chiffretexten als ein Spiel, das der Angreifer spielt. Der Vorteil des Angreifers wird angegeben als die Differenz der Wahrscheinlichkeit mit der der Angreifer c_b dem richtigen x_i zuordnen kann, und der Erfolgswahrscheinlichkeit eines zufällig ratenden Angreifers.

Sei $\mathcal{A}$ ein Angreifer. Dann lässt sich der Angriff als ein Spiel in zwei Schritten auffassen:

1. Im ersten Schritt erzeugt der Angreifer zwei Klartexte x_1, x_2.
2. Im zweiten Schritt erhält der Angreifer $\mathcal{A}$ das anzugreifende Chiffrat:
   Sei c_b die Verschlüsselung von einem der beiden Klartexte. Das Ziel von $\mathcal{A}$ ist nun den geheimen Wert b mit einer Wahrscheinlichkeit $> \frac{1}{2}$ zu ermitteln.

Dies führt zu dem Vorteil $\mathrm{Adv}_{\mathcal{A}}$ des Angreifers [Sma03, p.291]:

$$\mathrm{Adv}_{\mathcal{A}} = \left| Pr\Big(\mathcal{A}(\text{ermitteln}, c_b, k_p, x_1, x_2) = b\Big) - \frac{1}{2} \right|.$$

Dabei steht „Adv“ für Advantage. Weiter nennt man das Kryptosystem polynomiell sicher (engl. *polynomially secure*), wenn die Wahrscheinlichkeit des Ratens für Angreifer beschränkt ist durch:

$$\mathrm{Adv}_{\mathcal{A}} \leq \frac{1}{p(n)}.$$

Mit beliebigen Polynomen p vom Grad n. Man nennt den Vorteil des Angreifers dann vernachlässigbar klein.

Diese Formalisierung der Ununterscheidbarkeit von Chiffretexten kann erweitert werden auf die in Abschnitt 3.1 vorgestellten Angreifermodelle. Man spricht dann z.B. von einer Ununterscheidbarkeit von Chiffretexten bei einem Angreifer auf bekannten Klartext (engl.: *indistinguishability under chosen-ciphertext attack*/IND-CPA). Weiter: IND-CCA1, IND-CCA2 [BDPR, p.33]. Oft wird nur von semantischer Sicherheit eines Kryptosystems gesprochen. Hiermit ist IND-CPA-Sicherheit gemeint, da kein Angreifer mit Entschlüsselungsapparat betrachtet wird [DDN03, p.27].

3.2.4 Keine unautorisierte Verformbarbarkeit von Chiffretexten (non-malleability/NM)

Im Folgenden wird zur Definition von „keine unautorisierte Verformbarkeit von Chiffretexten“ (engl. *non-malleability*, kurz: NM) zum besseren Verständnis zunächst folgende intuitive Idee vorgestellt: Ein Kryptosystem das nicht NM-sicher ist, hat folgende Eigenschaft: Ein Angreifer kann einen Chiffretext c von Klartext x gezielt verformen, um einen daraus abgeleiteten Chiffretext c' zu erzeugen, welcher in einer ihm bekannten Beziehung zu c steht [Sma03, p.292].

Während die Ununterscheidbarkeit von Chiffretexten formalisiert, dass ein Angreifer aus einem Chiffretext nichts über den Klartext lernt, formalisiert NM, dass ein Angreifer aus einem Chiffretext keinen angeleiteten Chiffretext erzeugen kann.

Man betrachte das Szenario einer Versteigerung, welche die Partei $\mathcal{R}$ leitet. Teilnehmende Bieter sind die Parteien $\mathcal{S}$ und $\mathcal{A}$. Ihre Gebote geben sie an $\mathcal{R}$ verschlüsselt unter dem öffentlichen Schlüssel von $\mathcal{R}$ ab. Verwendet wird das Kryptosystem von Okamoto-Uchiyama. Dieses erlaubt eine homomorphe Addition zu einem Chiffretext mit dem öffentlichen Schlüssel. Dann könnte $\mathcal{A}$ folgenden Angriff durchführen:

1. $\mathcal{A}$ wartet bis $\mathcal{S}$ sein verschlüsseltes Gebot c abgibt. Der zugehörige Klartext x ist $\mathcal{A}$ nicht bekannt.
2. Dann fängt $\mathcal{A}$ das Gebot c von $\mathcal{S}$ ab und erzeugt ein eigenes Gebot c', dessen Klartext x' das um 1 erhöhte Gebot von x ist: $x' = x + 1$.

Somit gewinnt $\mathcal{A}$ die Auktion. Beispiel entnommen aus [KL14, p.388].

In der Literatur finden sich verschiedene formale Definitionen für NM-Sicherheit von [BDPR, p.34][DDN03, p.12] von denen Bellare et al. [BS99] beide als äquivalent nachgewiesen hat. Hier wird die Definition von Dolev [BDPR] vorgestellt:

Zunächst zur Notation: Sei $X \not\in \perp$ eine nicht-leere Menge von Klartexten und $x \in X$ ein zu X zugehöriger Klartext. Dann bezeichnet $X \leftarrow \mathcal{D}_{k_s}(C)$ die Entschlüsselung aller Chiffretexte $c \in C$. Es werden t-stellige Relationen betrachtet. Anstelle von $R(x_1, \ldots, x_t)$ wird $R(x, X)$ notiert, wo bei x ein gesonderter Klartext und X eine Menge mit $t-1$ Klartexten ist.

Der Angreifer hat nun zum Ziel zu einem Chiffretext c mit der Entschlüsselung x eine Menge abgeleiteter Chiffretexte C zu erzeugen, deren Entschlüsselung X zu x in einer Relation $\mathcal{R}(x, X)$ steht.

Das Sicherheitsziel wird durch ein Spiel beschrieben, das der Angreifer spielt. Dabei wird das Kryptosystem als sicher bewertet, wenn der Angreifer nicht besser abschneidet als ein ratender Angreifer. Die Differenz der Wahrscheinlichkeiten wird wie schon bei der Ununterscheidbarkeit von Chiffretexten als Vorteil des Angreifers angegeben.

Sei $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ ein Angreifer. Formal fasst man seine Aktivitäten in zwei Algorithmen $\mathcal{A}_1, \mathcal{A}_2$, die nacheinander ausgeführt werden. So lässt sich der Angriff als ein Spiel in zwei Schritten beschreiben:

1. Im ersten Schritt erzeugt der $\mathcal{A}_1$ eine Beschreibung des Klartextraums:
   Es wird ein asymmetrisches Kryptosystem angegriffen. Daher hat $\mathcal{A}_1$ Zugriff auf den öffent-

lichen Schlüssel k_p als Eingabeparameter. $\mathcal{A}_1$ erzeugt nun einen Samplingalgorithmus M aus dem Klartexte x gezogen werden können[3].

2. Im zweiten Schritt erhält $\mathcal{A}_2$ das anzugreifende Chiffrat :
   Aus M wird ein Klartext x gezogen, welches der $\mathcal{A}_2$ nicht kennt. Es wird ihm nur der zugehörige Chiffretext $c \leftarrow \mathcal{E}_{k_s}(x)$ gegeben. Die Herausforderung, welche $\mathcal{A}_2$ nun bewältigen muss, ist folgende: $\mathcal{A}_2$ erzeugt eine (Beschreibung der) Relation $\mathcal{R}$ und eine Menge $C \not\ni c$ von Chiffretexten. $\mathcal{A}_2$ hofft, dass für die zu C zugehörigen Klartexte X die Relation $R(x,X)$ gilt. Der $\mathcal{A}_2$ ist erfolgreich, wenn für sein erzeugtes C eine Relation $R(x,X)$ wahrscheinlicher ist als eine Relation $R(\tilde{x},X)$, wobei $\tilde{x} \in M$ ein zufällig gewählter Klartext ist.

   Die Wahrscheinlichkeit für das erfolgreiche Schätzen des Angreifers wird als $\mathrm{Succ}^{NM-a}_{A,K}(n)$ notiert. Die Wahrscheinlichkeit, dass die vom Angreifer erzeuge Relation R und Menge C zu einem zufälligen $\tilde{x} \in M$ passen, wird mit $\mathrm{Succ}^{NM-a}_{A,K,\$}(n)$ notiert.

Definition 13 (Keine unautorisierte Verformbarkeit: NM-CPA, NM-CCA1, NM-CCA2)**.** *Sei $K = (\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ ein asymmetrisches Kryptosystem und sei $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$ ein Angreifer. Für $a \in \{CPA, CCA1, CCA2\}$ und $n \in \mathbb{N}$ ist der Vorteil des Angreifers bei NM-a:*

$$\mathrm{Adv}^{NM-a}_{A,K}(n) = \left|\mathrm{Succ}^{NM-a}_{A,K}(n) - \mathrm{Succ}^{NM-a}_{A,K,\$}(n)\right|$$

mit

$$\mathrm{Succ}^{NM-a}_{A,K}(n) = Pr\Big((k_p,k_s) \leftarrow \mathcal{K};\ (M,s) \leftarrow A_1^{\mathcal{O}_1}(k_p);\ x \leftarrow M;\ c \leftarrow \mathcal{E}_{k_s}(x);\ (R,C) \leftarrow A_2^{\mathcal{O}_2}(M,c,s);\ \boldsymbol{x} \leftarrow \mathcal{D}_{k_s}(C) : c \notin C \wedge \perp \not\ni X \wedge R(x,X)\Big)$$

$$\mathrm{Succ}^{NM-a}_{A,K,\$}(n) = Pr\Big((k_p,k_s) \leftarrow \mathcal{K};\ (M,s) \leftarrow A_1^{\mathcal{O}_1}(k_p);\ x,\tilde{x} \leftarrow M;\ c \leftarrow \mathcal{E}_{k_s}(x);\ (R,C) \leftarrow A_2^{\mathcal{O}_2}(M,c,s);\ X \leftarrow \mathcal{D}_{k_s}(C) : c \notin C \wedge \perp \not\ni X \wedge R(\tilde{x},X)\Big).$$

Wobei je nach Angreifermodell für den Entschlüsselungsapparat $\mathcal{O}$ gilt:

$$\begin{aligned}
&\text{Wenn } a = CPA, \quad \text{dann ist } \mathcal{O}_1(\cdot) = \epsilon \qquad \text{und } \mathcal{O}_2(\cdot) = \epsilon.\\
&\text{Wenn } a = CCA1, \text{ dann ist } \mathcal{O}_1(\cdot) = \mathcal{D}_{k_s}(\cdot) \text{ und } \mathcal{O}_2(\cdot) = \epsilon.\\
&\text{Wenn } a = CCA2, \text{ dann ist } \mathcal{O}_1(\cdot) = \mathcal{D}_{k_s}(\cdot) \text{ und } \mathcal{O}_2(\cdot) = \mathcal{D}_{k_s}(\cdot).
\end{aligned}$$

Im Falle des CPA Angreifermodells gibt es keinen Entschlüsselungsapparat. Daher ist der Entschlüsselungsapparat das leere Wort. Im Falle des CCA1 Angreifermodells kann der Angreifer nach Analyse seiner Klartext-Chiffretext-Paare nicht erneut den Entschlüsselungsapparat anfragen, daher ist $\mathcal{O}_2$ das leere Wort. Im Falle des CCA2 Angreifermodells wird vorausgesetzt, dass der Angreifer den Entschlüsselungsapparat nicht anfragt den Chiffretext c zu entschlüsseln.

Bei dem Parameter s handelt es sich um Zustandsinformationen des Algorithmus $\mathcal{A}_1$ die an $\mathcal{A}_2$ weitergegeben werden kann (wie z.B. den öffentlichen Schlüssel). Bellare et al. benötigen diesen Pa-

[3] Dieser Schritt dient lediglich als Vorbereitung für den zweiten Schritt. Vergleiche hierzu den ersten Schritt des Angreifers in 3.2.3. Dort stellt dieser in der Vorbereitungsphase zwei Klartexte zur Verfügung. Von diesen wird eines dem Angreifer verschlüsselt zurückgegeben, welches er angreifen muss. Entsprechend kann hier der Angreifer mit M festlegen welche Klartextnachrichten benutzt werden um ein davon abhängiges Chiffrat zu erzeugen.

rameter für ihre Beweise von Implikationen zwischen den Sicherheitskriterien, welche im nächsten Abschnitt erläutert werden.

Sei die Rechenkapazität des Angreifers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ polynomiell durch $p(n)$ beschränkt, mit Polynom p und $n \in \mathbb{N}$. Ein Kryptosystem wird NM-a-sicher genannt ($a \in \{CPA, CCA1, CCA2\}$), wenn der Vorteil des Angreifers beschränkt ist durch:

$$\mathrm{Adv}_{A,K}^{NM-a}(n) \leq \frac{1}{p(n)}.$$

3.3 Implikationen zwischen den Sicherheitskriterien

Angreifermodelle und Sicherheitsziele wurden in vorherigen Abschnitten zu den Sicherheitskriterien IND-CPA/NM-CPA, IND-CCA1/NM-CCA1 und IND-CCA2/NM-CCA2 zusammengefasst. Da diese Definitionen teilweise Erweiterungen voneinander sind, existieren zwischen ihnen Implikationen: So impliziert CCA2-Sicherheit einen Schutz gegenüber dem schwächeren Angreifermodell CCA1, was direkt aus der Definition in 3.1 folgt.

Darüber hinaus gibt es weitere Implikationen, welche in [BDPR] alle bewiesen wurden. Dies führt zu folgenden Beziehungen zwischen den Sicherheitskriterien:

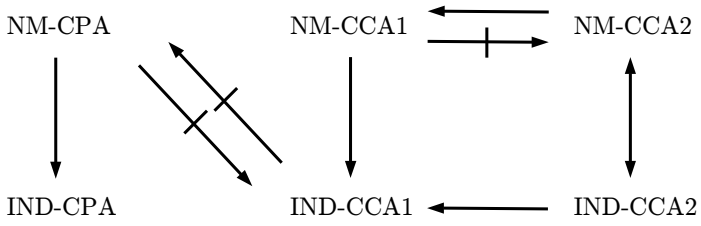

Abbildung 1: *Implikationen zwischen Sicherheitskriterien kryptografischer Verfahren. Ein Pfeil bedeutet, dass diese Implikation formal bewiesen wurde. Ein durchgestrichener Pfeil bedeutet, dass diese Implikation als ungültig bewiesen wurde.*

Unter dem gleichen Angreifermodell impliziert also NM-Sicherheit immer auch IND-Sicherheit.

3.4 Private Datenverarbeitung

Mehrere der in Kapitel 5 vorgestellten Studien setzen homomorphe Kryptosysteme zur Wahrung der Privatsphäre der hinterlegten Daten ein (engl. *privacy preserving*). Mit Datenverarbeitung unter Wahrung der Privatsphäre wird verstanden, dass die Daten unter einem Kryptosystem verschlüsselt sind, das eines der hier vorgestellten Sicherheitskriterien erfüllt.

3.5 Honest-but-curious Angreifermodell

Honest-but-curious (HBC) ist ein Angreifermodell für Protokolle in denen mehrere Parteien kooperieren. In einem Angriff werden eine oder mehrere Parteien, die am Protokoll teilnehmen, von

dem Angreifer übernommen. Der Angreifer hat dann Zugriff auf alle Daten, welche die Partei sehen kann. Der Angreifer ist ehrlich (engl. *honest*) und hält sich an das Protokoll. Er versucht jedoch die Daten, welche er lesen kann, aus Neugierde zu analysieren (engl. *curious*). Mehrere vom Angreifer übernommene Parteien können kooperieren, um Sicherheitsgarantien des Protokolls zu umgehen. Ist ein Protokoll sicher bzgl. des honest-but-curious Angreifermodells, werden folgende Annahmen gemacht:

1. Der Angreifer ist honest-but-curious.
2. Sei n die Anzahl der am Protokoll beteiligen Partien. Dann werden höchstens n/2 Parteien vom Angreifer übernommen.

In dieser Arbeit verfügen die honest-but-curious Angreifer darüber hinaus über polynomiell beschränkte Rechenkapazitäten. Damit sind sie nicht in der Lage die eingesetzten Kryptosysteme zu brechen. Neben honest-but-curious Angreifern betrachtet man *bösartige* Angreifer (engl. *malicious*). Bösartige Angreifer dürfen vom Protokoll abweichen und dürfen insbesondere keine Daten oder inkorrekte Daten senden [Sma03, p.385].

4 Sicherheit von semihomomorphen Kryptosystemen

In diesem Kapitel wird gezeigt, dass semihomomorphe Kryptosysteme nicht NM-sicher sein können. Anschließend wird erörtert, welche verbleibenden Sicherheitskriterien semihomomorphe Kryptosysteme erfüllen können.

4.1 Homomorphieeigenschaft impliziert Unautorisierte Verformbarkeit

In den vorangegangenen Kapiteln wurde bereits erwähnt, dass semihomomorphe Kryptosysteme nicht NM-sicher sein können. Dies wurde in der untersuchten Literatur stets informell begründet. Mit der formalen Definition eines semihomomorphen Kryptosystems, und des Sicherheitsziel NM, können wir im Folgenden einen Beweis für die Unerfüllbarkeit von NM-Sicherheit durch ein semihomomorphes Kryptosystemen formulieren.

Theorem 1. *Sei $K = (\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ein semihomomorphes Kryptosystem. Dann ist K nicht NM-a sicher ($a \in \{CPA, CCA1, CCA2\}$).*

Beweis. Sei $K = (\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ein semihomomorphes Kryptosystem und $k = (k_p, k_s) \in \mathcal{K}$ ein Schlüsselpaar des Kryptosystems. Dann ist die Verschlüsselungsfunktion $e_{k_s} : \mathcal{P} \to \mathcal{C}$ ein Gruppenhomorphismus zwischen den Gruppen $(\mathcal{P}, \oplus)$ und $(\mathcal{C}, \odot)$. Weiter sei $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ ein Angreifer.

$\mathcal{A}$ greift K wie folgt an:

1. $\mathcal{A}_1$ erzeugt einen Samplingalgorithmus M der beliebige Klartexte aus $\mathcal{P}$ erzeugen kann. $\mathcal{A}_1$ gibt den öffentlichen Schlüssel an $\mathcal{A}_2$ weiter: $s = \{k_p\}$.
2. $\mathcal{A}_2$ erhält den Chiffretext $c = e_{k_s}(x)$ von einem Klartext x den M erzeugt. Dann ist $x \in \mathcal{P}$ beliebig. Die weiteren Schritte beschreiben wie der Angreifer c verformt zu einem abgeleitetem Chiffretext c'. Dazu muss die Relation $\mathcal{R}$ und die Menge C angegeben werden (vgl. Definition 13).

   $\mathcal{A}_2$ wählt einen Klartext $y \in \mathcal{P}$ und erzeugt den Chiffretext $d = e_{k_p}(y)$. Nun verknüpft der Angreifer die beiden Chiffretexte über die abgeschlossene Verknüpfung $\odot$ auf $\mathcal{C}$: $c' = c \odot d$.

   Da e_{k_p} ein Homomorphismus ist, gilt:

$$c' = c \odot d = e_{k_p}(x) \odot e_{k_p}(y) \overset{Hom.}{=} e_{k_p}(\underbrace{x \oplus y}_{=:x'}) = e_{k_p}(x')$$

Jetzt definiert $\mathcal{A}_2$ die Mengen $C := \{c'\}$ und $X := \{x'\}$. Die Elemente der Relation $\mathcal{R}$ sind zweistellige Tupel. $\mathcal{A}_2$ definiert die Relation wie folgt: $(a,b) \in \mathcal{R}$ genau dann, wenn $a = x$ und $b = x \oplus y$.

Da $\mathcal{A}_2$ den Klartext y selber wählt, ist die Wahrscheinlichkeit, dass die Relation $\mathcal{R}(x,X) = \mathcal{R}(x, x \oplus y)$ gilt gleich 1.

Sei nun $\tilde{x} \neq x$ ein weiterer Klartext welchen der Samplingalgorithmus M erzeugt. Damit die Relation $\mathcal{R}(\tilde{x}, x \oplus y)$ erfüllt ist muss gelten:

$$x \oplus y \stackrel{!}{=} \tilde{x} \oplus y$$

Da $(\mathcal{P}, \oplus)$ Gruppe, existiert das eindeutige Inverse y^{-1} von y.

$$\begin{aligned} \Leftrightarrow \quad & x \oplus e_{\mathcal{P}} = \tilde{x} \oplus e_{\mathcal{P}} \\ \Leftrightarrow \quad & x = \tilde{x} \qquad \lightning \end{aligned}$$

Damit ist die Relation $\mathcal{R}(\tilde{x}, x \oplus y)$ nicht erfüllt und der Vorteil des Angreifers ist 1-0=1. Das heißt, der Angreifer kann bei dem semihomomorphen Kryptosystem K den abgeleiteten Chiffretext c' mit hundertprozentiger Wahrscheinlichkeit erzeugen. Sein Vorteil ist nicht vernachlässigbar klein.

In dem vorgestellten Angriff benötigt $\mathcal{A}$ keinen Entschlüsselungsapparat um die Menge $C = \{c'\}$ und Relation $\mathcal{R}$ zu erzeugen. Der Angreifer erzeugt C mit der Verknüpfung $\odot$ im Chiffreraum und nutzt lediglich die Homomorphieeigenschaft der Verschlüsselungsfunktion aus. Daher ist der Angriff erfolgreich für alle drei Angreifermodelle NM-CPA, NM-CCA1 und NM-CCA2. □

4.2 Angreifermodelle bei semihomomorphen Kryptosystemen

Da semihomomorphe Kryptosystem nach Kapitel 2.4 asymmetrisch sind, hat der Angreifer immer die Option beliebige ausgewählte Klartexte anzugreifen und Klartext-Chiffretext-Paare zu erzeugen. Daher müssen homomorphe Kryptosysteme *mindestens* CPA-Sicherheit gewährleisten, d.h. sie müssen sicher sein gegenüber Angreifern, welche dem CPA Angreifermodell entsprechen.

Die Frage welche sich jetzt stellt, ist, ob semihomomorphe Kryptosysteme gegenüber den stärkeren Angreifermodellen CCA1 und CCA2 sicher sein können?

Eine Folgerung von Theorem 1 und den Implikationen zwischen den Angreifermodellen in 3.3 ist, dass ein semihomomorphes Kryptosystem *nicht IND-CCA2-sicher* sein kann. Den IND-Sicherheit unter dem CCA2 Angreifermodell würde NM-Sicherheit unter dem CCA2 Angreifermodell implizieren.

Damit ist IND-CCA1 das stärkste mögliche Angreifermodell, gegen welches semihomomorphe Kryptosysteme sicher sein können. IND-CPA ist das schwächste Angreifermodell, gegen das sie sicher sein müssen.

Es existieren semihomomorphe Kryptosysteme, die eines dieser Sicherheitskriterien erfüllen. In [Pai99, p.231] wird die IND-CPA Sicherheit von Paillier und in [WS08, p.7] die IND-CCA1-Sicherheit von ElGamal nachgewiesen.

5 Klassifikation Homomorpher Kryptosysteme

In diesem Kapitel werden verschiedene Anwendungen vorgestellt, in denen ein oder mehrere semihomomorphe Kryptosysteme zum Einsatz kommen.

Für eine Klassifizierung erfolgt die Untersuchung der Anwendungen unter folgenden Kriterien:

1. Es sollen die *Eigenschaften eines semihomomorphen Kryptosystems identifiziert* werden, welches Wissenschaftler zu deren Einsatz bewegt hat.
2. Es wird untersucht, welche *Funktionen und Operatoren* mit der homomorphen Verknüpfung eines Kryptosystems realisiert werden.
3. Die verwendeten Kryptosysteme können nur IND-CPA oder IND-CCA1 sicher sein. Sie sind insbesondere nicht NM-sicher. Es wird ausgeführt, inwiefern dies bei der Implementierung berücksichtigt wird, um die *Integrität* von Rechenoperationen zu gewährleisten.

5.1 Autocrypt [TSCS13]

Server sind ständig durch Angriffe bedroht, die bis hin zu ihrer kompletten Übernahme führen können. Um Datendiebstahl und Vertraulichkeitsverletzungen vorzubeugen, ist es ratsam nur mit verschlüsselten Datenbeständen auf den Servern zu arbeiten. Die Anpassung der Programme, um mit verschlüsselten Daten zu arbeiten, wollen die Wissenschaftler automatisieren, indem sie die Arbeit der Programmtransformation mit einem selbst entwickelten Compiler abwickeln. Das Ergebnis ihrer Forschung ist der Compiler Autocrypt.

Ein Server läuft als nicht vertrauenswürdige virtuelle Maschine (VM). Inhalte werden außerhalb der VM auf einem vertrauenswürdigen Schlüsselserver verschlüsselt. Autocrypt bestimmt automatisch welche Verschlüsselung des Wertes einer Variable für Rechenoperationen notwendig sind und konvertiert ihren Wert im Programmablauf durch Einfügen von Hypercalls[1]. In Abhängigkeit davon, welche mathematische Grundoperation mit einer Variable ausgeführt wird, rechnet das transformierte Programm mit der passenden Darstellung im semihomomorphen Kryptosystem. Wenn im Ursprungscode Additionen von zwei Variablen durchgeführt werden, werden ihre Wert mit Paillier[2] verschlüsselt. Wird das Ergebnis allerdings später multipliziert, dann muss der Wert einer Variable zur Laufzeit konvertiert werden nach ElGamal[3]. Veranschaulicht wird dies in Abbildung 2. Durch die Konvertierung der Werte zwischen Darstellungen in Paillier und ElGamal wird die Verwendung eines langsameren vollhomomorphen Kryptosystems vermieden.

[1] Ein Hypercall ist vergleichbar mit einem Systemaufruf in der Virtualisierungsumgebung. Ein Hypercall ermöglicht die Ausführung von Instruktionen die höhere Rechte benötigen. https://wiki.xenproject.org/wiki/Hypercall
[2] Paillier ist ein additiv semihomomorphes Kryptosystem [Pai99].
[3] ElGamal ist ein multiplikativ semihomomorphes Kryptosystem [YPB14, p.32].

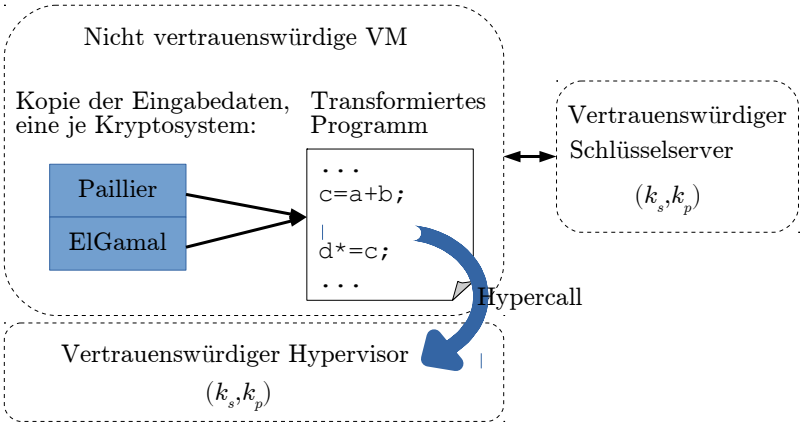

Abbildung 2: *Die nicht vertrauenswürdige VM führt an einer Stelle im Programm einen Hypercallaufruf aus, um den Wert in der Variable c von Paillier nach ElGamal zu konvertieren. Alle Eingabedaten des Programms liegen in der VM verschlüsselt vor. Die Konvertierung wird von dem vertrauenswürdigen Hypervisor ausgeführt.*

Bei der Entwicklung von Autocrypt ist das primäre Ziel alle Rechenoperationen unter Wahrung der Privatsphäre durchzuführen. Die Forscher erwähnen explizit, dass es nicht Ziel ist die Integrität der verarbeiteten Daten eines transformierten Programms sicherzustellen.

Zur Sicherheit des transformierten Programms: Es wird angenommen, dass die zu transformierenden Programme keine Größen von verarbeitenden Daten verbergen müssen. Man geht davon aus, dass im zu transformierenden Programm Maßnahmen gegen Seitenkanalangriffe umgesetzt worden. Es wird jedoch nicht darauf eingegangen, inwiefern dieser Schutz unter der Transformation durch Autocrypt erhalten bleibt. Die semantische Sicherheit von Paillier und ElGamal wird genutzt, um eine Sicherheit gegenüber dem IND-CPA Angreifermodell für alle Operationen auf der VM garantieren. Hierzu skizzieren die Autoren einen Beweis.

Autocrypt realisiert homomorphe Operation auf Bits durch Verwendung des Paillier Kryptosystems für die Bitdarstellung von Integern anstelle des Integerwerts selbst. Wie Autocrypt mit Fließkommazahlen umgeht wird nicht erwähnt.

Klassifizierungskriterien:

1. Die semantische Sicherheit der semihomomorphen Kryptosysteme von Paillier und ElGamal ist Grundlage um von Autocrypt transformierte Programme sicher gegenüber IND-CPA Angreifern zu machen. Es wird eine Kombination aus additiven und multiplikativen semihomomorphen Kryptosystemen für Rechenoperationen im transformierten Programm verwendet, da diese effizienter sind als untersuchte vollhomomorphe Kryptosysteme.
2. Das Paillier Kryptosystem wird eingesetzt, um homomorphe Bitoperationen zu ermöglichen.
3. Eine Integrität der Rechenoperationen wird ausdrücklich nicht berücksichtigt.

5.2 Machine Learning Classification over encrypted data [BPTG15]

Es wird ein Privatsphäre wahrendes[4] Verfahren des Maschienenlernens entworfen, bei dem sowohl die zu klassifizierenden Daten (zusammengefasst in Merkmalsvektoren[5]) als auch die Klassifizierermodelle[6] vertraulich bleiben. Dazu wird eine Bibliothek konstruiert, aus der modular beliebige Privatsphäre wahrende Klassifizierer erstellt werden können.

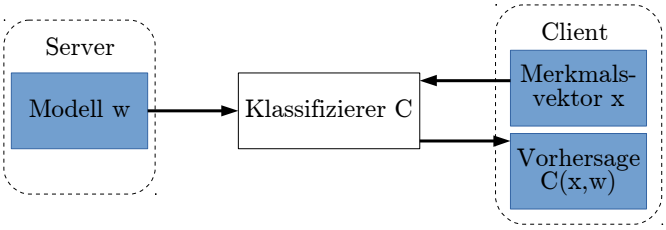

Abbildung 3: *Der nicht vertrauenswürdige Klassifizierer enthält verschlüsselte Eingaben vom Server und Client. Nach der Klassifizierung darf der Server nichts über die Merkmalsvektoren vom Clienten und der Client darf nichts über das Modell für die Klassifizierung vom Server lernen.*

In dem Design dieser Module werden homomorphe Kryptosysteme verschieden eingesetzt:

Für die Klassifizierung müssen die Module Vergleichsoperationen $(<, \leq)$ zwischen verschlüsselten Werten durchführen können. Die Autoren verwenden hierzu ein Protokoll von Veugen [Veu11], dass mit jedem semantisch sicheren homomorphen Kryptosystem verwendet werden kann. Veugen selbst evaluiert die Performance des Protokolls unter Verwendung der Kryptosysteme von Goldwasser-Micali[7] und Paillier. Da die Autoren dieser Studie keine weiteren Gründe für den Einsatz gerade dieser Kryptosysteme aufführen, wird angenommen, dass sie sich an dem Beispiel von Veugen orientieren.

Weiter sollen die Klassifizierer Polynome evaluieren können, die einen binären Entscheidungsbaum repräsentieren (siehe Abbildung 4): Die Blätter des Entscheidungsbaums sind mögliche Klassen c, zu denen der Merkmalsvektor x zugeordnet werden kann. Den Weg im Entscheidungsbaum zu gehen, entspricht der privaten Evaluation des Polynoms, dass diesen Entscheidungsbaum repräsentiert. Für die Evaluation des Entscheidungsbaums geben die Autoren eine Tiefe von $\log_2 \cdot h_{max}$ Multiplikationen an. Durch diese beschränke Tiefe von Rechenoperationen eignet sich der der Einsatz des eingeschränkten (hier: leveled) vollhomomorphen Kryptosystems von Brakerski-Gentry-Vaikuntanathan[8] (BGV). Das BGV Kryptosystem ist IND-CPA sicher [BGV12].

Als Vorteil des Einsatzes von Paillier wird ein großer Klartextraum von 2^{1024} Bit genannt. Die Größe des Klartextraums wird genutzt, um Fließkommazahlen zu verschlüsseln, indem diese durch Multiplikation mit großen Exponenten zu Integern konvertiert werden.

[4] privacy-preserving, siehe: 3.4
[5] Ein Merkmalsvektor ist eine einheitliche Darstellung des zu klassifizierenden Objektes.
[6] Der Klassifizierer C benutzt das Klassifizierermodell w um den Merkmalsvektor x einer Klasse c=C(x,w) zuzuordnen.
[7] Das Kryptosystem von Goldwasser-Micali erlaubt eine homomorphe bitweise XOR Verknüpfung. [GM82]
[8] Die Autoren verwenden HELib, eine Bibliothek die das Kryptosystem von Brakerski-Gentry-Vaikuntanathan implementiert.

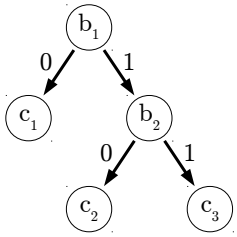

Abbildung 4: *Damit der Klassifizierer in einem binärem Baum privat einen Pfad von der Wurzel zum Blatt geht, wird dieser als ein Polynom repräsentiert, das privat berechnet wird. Der Wert des Knotens $b_i \in \{0,1\}$ bestimmt, welchen Pfad entlang gegangen wird und ist von einem Vergleich aus Einträgen des Modells w und Merkmalsvektors x abhängig. z.B. Falls $x_1 \leq w_1 \rightarrow b_1 = 1$. Das zu diesem Baum zugehörige Polynom ist: $P(b_1,b_2,c_1,c_2,c_3) = b_1(b_2 \cdot c_3 + (1-b_1)c_2) + (1-b_1)c_2$.*

Ein weiterer Einsatz von Paillier erfolgt in der Berechnung eines privaten Skalarproduktes zwischen zwei Parteien:

Gegeben seien die Vektoren $x = (x_1,\ldots,x_n)$ von A und $y = (y_1,\ldots,y_n)$ von B, wobei alle Einträge Klartexte sind. Sei $k = (k_s,k_p)$ das Schlüsselpaar im Paillier-Kryptosystem:

1. B verschlüsselt die Komponenten y_i mit k_s und sendet $e_{k_s}(y_i)$ an A.
2. A berechnet $x_i \cdot y_i$ durch Potenzierung des Chiffretextes von B mit x_i. Dies funktioniert genauso wie im Okamoto-Uchiyama Kryptosystem in Abschnitt 2.4.4. Unterschied ist lediglich der Modulus[9] von n^2 bei Paillier mit $n = pq$. Durch Multiplikation der Chiffretexte wird dann die i-te Komponente im Klartext aufsummiert:

$$e_{k_s}(\langle x,y\rangle) = \prod_i e_{k_s}(y_i)^{x_i} \bmod n^2.$$

Klassifizierungskriterien:

1. Die semihomomorphen Kryptosysteme von Goldwasser-Micali und Paillier werden verwendet, um verschlüsselte Werte bzgl. ihrer numerischen Größe zu vergleichen. Der große Klartextraum von Paillier wird genutzt, um Fließkommazahlen zu verschlüsseln. Das beschränkte vollhomomorphe Kryptosystem von Brakerski-Gentry-Vaikuntanathan wird eingesetzt, um Polynome zu evaluieren.
2. Es wird ein privates Skalarprodukt auf Basis von Paillier implementiert.
3. Der Einsatz von homomorphen Kryptosystemen geschieht nur unter dem Hintergrund die Daten vertraulich zu verarbeiten. Es wird weder die Möglichkeit einer unautorisierten Verformbarkeit ausdrücklich erwähnt, noch in dem Design der Module zur Konstruktion von Klassifizierern berücksichtigt. Die Autoren gehen jedoch von einem honest-but-curious Angreifermodell aus, d.h. es wird angenommen, dass ein Angreifer jegliche Kommunikation mitlesen kann, sich jedoch protokollkonform verhält. Insbesondere kann er nicht die Verschlüsselung brechen. Da die Eingabedaten des Klassifizierers verschlüsselt sind, kann der Angreifer keine Einsicht in ihre Inhalte bekommen.

[9] Bei Okamoto-Uchiyama ist der Modulus $n = p^2q$. In beiden Fällen sind p,q Primzahlen.

5.3 Privacy Preserving Matrix Factorization [NIW+13]

Für die Generierung von personenspezifischen Empfehlungen wird Matrixfaktorisierung verwendet [KBV09]. Die Matrixfaktorisierung ist Bestandteil eines Empfehlungssystems (engl. *recommender system*), das anhand vorheriger Bewertungen die Personen für Objekte abgegeben haben, zukünftige Empfehlungen geben kann (z.B. Einkäufe in einem Onlineladen).

In dieser Studie möchten die Wissenschaftler ein Empfehlungssystem entwerfen, dass die Matrizenfaktorisierung unter Wahrung der Privatsphäre der abgegebenen Bewertungen durchführt. Das Empfehlungssystem soll Empfehlungen geben können ohne die abgegebenen Bewertungen einer Personen zu lernen oder die Objekte welche Personen bewertet haben. Es werden nur die Objektkategorien gelernt.

Die Matrixfaktorisierung wird von dem Empfehlungssystem in Kooperation mit einem Schaltkreisanbieter durch Anwendung des „Garbled Circuit" Protokolls von Yao [Yao82] durchgeführt (siehe Abbildung 5). Es wird nun die Kommunikation zwischen Empfehlungssystem und Schaltkreisanbieter erläutert: Das Empfehlungssystem erhält mit Hash-ElGamal[10] verschlüsselte Bewertungen r_{ij} von Personen i für Objekte j, mit denen es die Matrixfaktorisierung durchführen möchte. Dazu übergibt das Empfehlungssystem dem Schaltkreisanbieter die Bewertungen und Spezifikationen zum Erstellen des Schaltkreises (u.a. Gesamtzahl von Objekten, Anzahl bewerteter Objekte). Dann erhält das Empfehlungssystem vom Schaltkreisanbieter einen Schaltkreis mit dem die Matrixfaktorisierung ausgeführt werden kann.

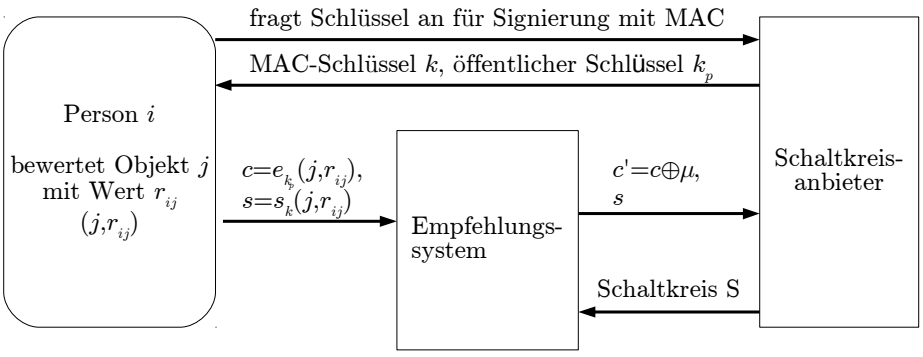

Abbildung 5: *Ablauf zur Erstellung des Schaltkreises der die Matrixfaktorisierung berechnet.*

Die Autoren betrachten zwei Angreifermodelle: Im ersten Fall können das Empfehlungssystem und der Schaltkreisanbieter von einem honest-but-curious Angreifer übernommen werden. Im zweiten Fall wird nur das Empfehlungssystem von einem bösartigen Angreifer übernommen, der sich nicht an das Protokoll hält. In Abbildung 5 wird nur das Design beim stärkeren Angreifermodelle betrachtet: Damit das Empfehlungssystem die Wertepaare (j,r_{ij}) nicht lernt, werden diese von den Personen i unter dem öffentlichen Schlüssel k_p des Schaltkreisanbieters verschlüsselt zu $c = e_{k_s}(j,r_{ij})$.

[10] Hash-ElGamal ist ein semihomomorphes Kryptosystem zur XOR Verknüpfung [NIW+13, Appendix B] und IND-CPA sicher [CMPP06, p.6].

Da das Empfehlungssystem nicht möchte, dass der Schaltkreisanbieter die Wertepaare lernt, werden diese bitweise mit Zufallswerten μ maskiert zu $c' = c \oplus \mu$. Ein Angreifer könnte Wertepaare verändern oder alle Wertepaare bis auf eines durch Dummywertepaare ersetzen (z.B. alle bewerten das gleiche Modell). Dann gibt der Schaltkreis bei der Matrixfaktorisierung Auskunft welches Objekt eine Person bewertet hat. Damit dies nicht möglich ist, werden Wertepaare vor dem Verschlüsseln von den Personen mit einem MAC-Code signiert, den der erstellte Schaltkreis als zusätzliche Eingabe benötigt. Dazu übergibt der Schaltkreisanbieter der Person einen neuen MAC-Schlüssel je Wertepaar. Der Schaltkreis wird so konstruiert, dass er zu 0 evaluiert falls der Angreifer Wertepaare verändern sollte.

Klassifizierungskriterien:

1. Es wird ein IND-CPA sicheres semihomomorphes Kryptosystem zur XOR Verknüpfung verwendet. Die Autoren entscheiden sich aus Effizienzgründen für die Verwendung von Hash-ElGamal gegenüber dem bekannteren Paillier Kryptosystem. Die Implementierung von Hash-ElGamal ist effizient im Vergleich zu Paillier, dies wird jedoch nicht ausgeführt und auch nicht praktisch verglichen.
2. Hash-ElGamal wird zur Verschlüsselung und Maskierung durch Addition von Zufallswerten verwendet.
3. Die Autoren berücksichtigen einen bösartigen Angreifer, der verschlüsselte Wertepaare willkürlich verändern oder komplett ersetzten könnte, indem er selber Wertepaare erzeugt. Dies wird verhindert, indem Wertepaare von den Personen vor dem Verschlüsseln mit einem MAC-code signiert werden.

5.4 Efficient and Secure Comparison for On-Line Auctions [DGK07]

Damgard et al. motivieren ihre Veröffentlichung damit, dass Vergleiche von verschlüsselten Zahlenwerten vielfältig in privaten verarbeitenden Protokollen benötigt werden. Um Vergleiche höchst performant durchführen zu können, haben sie ein eigenes additiv semihomomorphes Kryptosystem entwickelt (im Folgenden DGK genannt), das IND-CPA sicher ist.

Sie stellen ihr Protokoll für den Fall eines Auktionshauses vor, bei ein aktuelles Höchstgebot x mit einem privaten Gebot m eines Bieters verglichen werden soll. Das aktuelle Höchstgebot x ist öffentlich bekannt. Das Gebot m soll beim Vergleich privat bleiben, damit das Auktionshaus die Preise nicht beliebig erhöhen kann. Denn würde m>x gelten, so wird x nur solange inkrementell erhöht, bis m das Höchstgebot unter allen Bietern ist. Dieses Schema entspricht dem Bieterverfahren bei Ebay.

Für die Kategorisierung werden besondere Designeigenschaften des DGK Kryptosystems untersucht, die das Protokolls ausnutzt: Damit Rechenoperationen bei dem DGK Kryptosystem mit wenig Rechenaufwand erfolgen, soll der Klartextraum $\mathbb{Z}_u$ möglichst klein sein. Sei l die Bitlänge einer zu verschlüsselnden Zahl $m = m_l \dots m_1$. Dann wählt das Kryptosystem u als die kleinste Primzahl größer als $l + 2$. Seien nun r, r' Zufallszahlen[11] und (n, g, h, u) der öffentliche Schlüssel des DGK Kryptosystems. Dann ist eine Eigenschaft des Kryptosystems, dass bei der homomorphen Addition modular reduziert wird bzgl. u:

[11] vgl. mit der Verschlüsselungsfunktion von Okamoto-Uchiyama in 2.4.4

$$e_{k_p}(m,r) \cdot e_{k_p}(m',r') \bmod n = e_{k_p}\ (m + m' \bmod r + r')$$

Da das verwendete Protokoll viele Addition mod u durchführt, soll u möglichst klein sein, aber immer noch genug groß, damit eine modulare Reduktion vermieden wird.

Bei Paillier ist die modulare Reduktion im Klartextraum mit $n = pq$ sehr groß, weil p,q große Primzahlen sind:

$$e_{k_p}(m,r) \cdot e_{k_p}(m',r') \bmod n^2 = e_{k_p}(m + m' \bmod n, r + r')$$

Die Autoren beweisen, dass ihr Kryptosystem IND-CPA sicher ist.

Klassifizierungskriterien:

1. Es wurde ein neues Kryptosystem mit kleinen Klartextraum speziell für das Vergleichsprotokoll erstellt um noch schneller private Zahlenvergleiche durchführen zu können als bisherige Umsetzungen.
2. Es werden keine weiteren homomorphen Operationen simuliert, die über die Additivität des DGK Kryptosystems hinausgehen.
3. Die Autoren gehen von einem honest-but-curious Angreifer aus. Eine unautorisierte Verformbarkeit von Chiffraten wird nicht untersucht.

5.5 Fingerprinting Protocol for Images Based on Additive Homomorphic Property [KT05]

Die Autoren entwerfen ein neues Verfahren, um asymmetrisch Fingerabdrücke in verschlüsselten Bildern zu hinterlegen. Sie benutzen dazu das additiv semihomomorphe Kryptosystem von Okamoto-Uchiyama, um eine möglichst große Verschlüsselungsrate zu erreichen. Bisherige Ansätze um asymmetrisch Wasserzeichen einzubetten benötigten bei einer Datengröße von 1MB bis zu 1GB Kommunikationsdaten. Es wird erwähnt, das der große Klartextraum von Okamoto-Uchiyama eine größere Verschlüsselungsrate ermöglicht.

Um die Einbettung möglichst robust gegenüber Manipulationen zu machen, sollen die Fingerabdrücke im Frequenzraum[12] eines Bildes eingebettet werden. Die reellen Zahlen des transformierten Bildes werden zu Integern quantisiert um das Bild verschlüsseln zu können.

Die Einbettung eines Fingerabdrucks erfolgt wie in Abbildung 6 veranschaulicht:

1. Der Kunde erzeugt einen Fingerabdruck, verschlüsselt ihn mit seinem öffentlichen Schlüssel und sendet den Chiffretext an den Händler. Mit einem Zero-Knowledge-Proof[13] wird dem Händler nachgewiesen, dass der Chiffretext tatsächlich einen nutzerspezifischen Fingerabdruck enthält.

[12] Das „klassische“ RGB Bild ist eine Linearkombination von ortsabhängigen Intensitäten - d.h. Pixeln mit einem Integerwert für die Intensität der Farbe. Man spricht von einer Darstellung im Ortsraum. Eine Fouriertransformation wechselt die Basis unter der das Bild dargestellt wird in den Frequenzraum. Das transformierte Bild wird dargestellt mittels einer Basis aus periodischen Funktionen.

[13] Ein Zero-Knowledge-Proof ermöglicht einer Partei A, einer anderen Partei B nachzuweisen das eine Aussage stimmt, ohne mehr als das offenlegen zu müssen (auch: engl. minimal disclosure) [BCC88].

2. Nun verschlüsselt der Händler sein digitales Bild unter dem öffentlichen Schlüssel des Kunden und bettet den Fingerabdruck durch homomorphe Verknüpfung ein.
3. Der Kunde entschlüsselt das Bild mit dem eingebetteten Fingerabdruck, ohne jedoch in der Lage zu sein diesen zu entfernen.

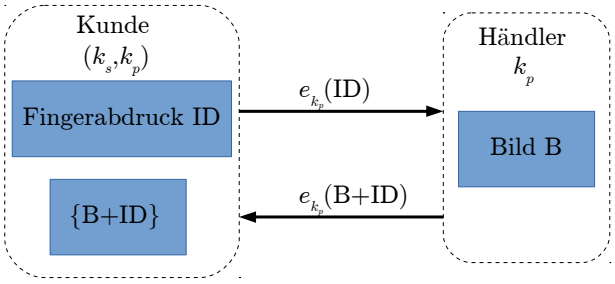

Abbildung 6: *Ablauf der Einbettung eines nutzerspezifischen Fingerabdrucks. Die Einbettung erfolgt in dem verschlüsselten digitalen Bild, damit nur der Kunde das Bild mit Fingerabdruck erhalten kann. Würde der Händler Zugriff auf das Bild mit Fingerabdruck haben, könnte er es selber vervielfältigen.*

Klassifizierungskriterien:

1. Es wird die Asymmetrie des Kryptosystems genutzt um asymmetrische Fingerprints zu erzeugen. Wären die Fingerprints symmetrisch, könnte der Händler den Fingerprint selber einbetten, davon Kopien verbreiten und Kunden hintergehen.
   Die Autoren entscheiden sich für das Okamoto-Uchiyama Kryptosystem anstelle von Paillier, weil es weniger Rechenschritte benötigt.
2. Es werden keine weiteren homomorphen Operationen simuliert, die über die Additivität des Okamoto-Uchiyama Kryptosystems hinausgehen.
3. Die Autoren erwähnen die Möglichkeit einer unautorisierten Verformbarkeit der Chiffrate, ohne dies bei der Sicherheit ihres Protokolls berücksichtigen zu müssen: Sicherheitsziele in ihrem Verfahren sind, dass der Kunde nicht das Wasserzeichen entfernen kann und der Händler nicht den Fingerabdruck des Kunden entschlüsseln kann. Kunde und Händler weichen nicht vom Protokoll ab (honest-but-curious).

5.6 Privacy Preserving Face Recognition [EFG+09]

Erkin et al. stellen ein Privatsphäre wahrendes Gesichtserkennungssystem vor, bei dem sowohl die Eingabebilder, als auch das Ergebnis ihrer Analyse vom Server nicht im Klartext einsehbar sind. Der Analyse liegt ein Eigenface Algorithmus zugrunde, welcher auf verschlüsselten Bildern arbeitet. Eingesetzt werden die Kryptosysteme Paillier und DGK, welches auch in 5.4 zum Einsatz kommt.

Bei der Schlüsselgenerierung von Paillier wird eine Optimierung aus [DJ01, p.16] verwendet um den Parameter g des öffentlichen Schlüssels (n,g) zu finden. Sei n wieder $n = pq$ für zwei Primzahlen, dann müsste g nach Paillier [Pai99] so gewählt werden, dass n die Ordnung von g teilt. Die Optimierung setzt $g = n+1$ und beschleunigt so die Verschlüsslung.

DGK wird aus Performancegründen zusätzlich zu Paillier eingesetzt. Die Autoren geben dabei wie in 5.4 den kleinen Klartextraum an. Da die Exponenten kleiner sind, ist die Verschlüsselung effizienter.

In den Verschlüsselungsfunktion beider Kryptosysteme werden vom Klartext unabhängige Parameter im Voraus berechnet, um die Verschlüsselung zu beschleunigen. Sei (n,g) der öffentliche Schlüssel unter Paillier und (n',g',h',u') der öffentliche Schlüssel im DGK Kryptosystem. Dann wird der Klartext x mit Zufallszahlen r,r' verschlüsselt zu:

$$\text{Paillier: } c = g^x r^n \bmod n^2, \qquad \text{DGK: } c = g'^x h'^{r'} \bmod n'$$

In beiden Kryptosystemen können die Faktoren r^n und $h'^{r'}$ bereits im Voraus berechnet werden.

Featurevektoren werden im Algorithmus diskretisiert, indem auf die nächste Ganzzahl gerundet wird, da die Kryptosysteme Ganzzahlen verschlüsseln.

Rechenoperationen im Chiffreraum des privaten Gesichtserkennungssystems:

Der Eigenface Algorithmus ist trotz der Implementierung auf verschlüsselten Daten vertretbar in der Performance, d.h. der Algorithmus ist in der Lage ein verschlüsseltes Bild von 92×112 Pixeln mit 320 Gesichtstemplates der Datenbank in vierzig Sekunden zu vergleichen. Verwendet wird ein Computer mit einem 2.4 GHz AMD Opteron Dualcore Prozessor und 4GB Arbeitsspeicher.

Da eine schnelle Berechnung von komplexen Funktionen auf verschlüsselten Daten nicht selbstverständlich ist, wird die Umsetzung der Rechenoperationen des Eigenface Algorithmus im Folgenden untersucht: Ein in eckige Klammern gesetztes Element bedeutet, dass es mit Alices öffentlichen Schlüssel verschlüsselt ist, die ein Gesicht analysieren möchte. Sie übergibt das verschlüsselte Gesicht und ihren öffentlichen Schlüssel an Bob, der dank homomorpher Kryptographie in der Lage ist den Algorithmus durchzuführen ohne das Gesicht direkt zu sehen.

- *Projektion* des verschlüsselten Eingabebildes $[\![\Gamma]\!]$ auf die Basis von Eigenfacevektoren $u_1,\ldots,u_K$. Die Eigenfacevektoren von Bob sind aus Trainingsdaten entstanden und werden als private Daten betrachtet, die Alice nicht lernen darf. Bei der Projektion wird mit der gleichen Technik wie in 5.2 ein Skalarprodukt durch Potenzieren berechnet. Das Ergebnis ist ein verschlüsselter Featurevektor des Eingabebildes $[\![\overline{\Omega}]\!]$, welcher nun mit Featurevektoren der Datenbank verglichen werden kann um das Gesicht zuzuordnen.
- *Abstand D* von Featurevektoren $\{\Omega_1,\ldots,\Omega_M\}$ der Datenbank von Bob zum Featurevektor des Eingabebildes $[\![\overline{\Omega}]\!]$. Da man nur an der relativen Ordnung der Abstände interessiert ist, genügt der Vergleich der quadierten Abstände:

$$\begin{aligned} D(\Omega,\overline{\Omega}) &= \|\Omega-\overline{\Omega}\|^2 = (\omega_1-\overline{\omega}_1)^2+\ldots+(\omega_K-\overline{\omega}_K)^2 \\ &= \underbrace{\sum_{i=1}^{K}\omega_i^2}_{S_1} + \underbrace{\sum_{i=1}^{K}(-2\omega_i\overline{\omega}_i)}_{S_2} + \underbrace{\sum_{i=1}^{K}\overline{\omega}_i^2}_{S_3} \end{aligned} \tag{5.1}$$

  Nun werden die drei Summenblöcke getrennt zur Berechnung des verschlüsselten Abstands $[\![D(\Omega,\overline{\Omega})]\!]$ verwendet: Da Bob den Server betreibt, kennt er die Komponenten ω_i der Featurevektoren in der Datenbank und kann S_1 direkt im Klartext berechnen und *anschließend*

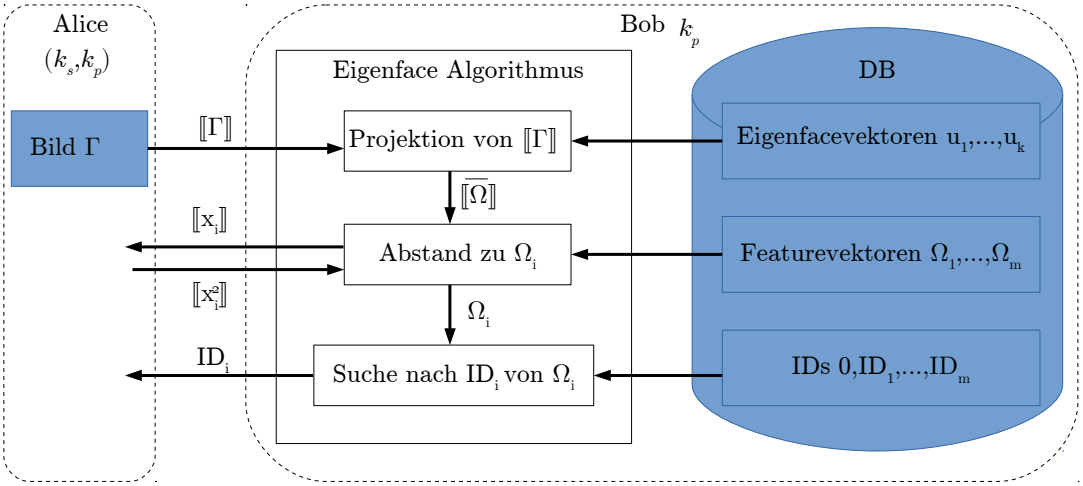

Abbildung 7: *Alice möchte die ID (das Gesicht einer Person) wissen. Der Algorithmus extrahiert das Gesichtes aus dem Bild und gleicht es mit der Datenbank von Bob ab. Bob kann den Algorithmus größtenteils alleine ausführen, benötigt jedoch einmal die Hilfe von Alice um ein Zwischenergebnis zu quadrieren.*

mit Alices öffentlichen Schlüssel verschlüsseln. Da er die Komponenten $\overline{\omega}_i$ des Featurevektor vom Eingabebild nur verschlüsselt vorliegen hat, muss er S_2 analog wie beim Skalarprodukt in 5.2 durch potenzieren berechnen. Letztendlich kann Bob S_3 nur in Kooperation mit Alice berechnen, da ihm bei beide Faktoren des Produkts unbekannt sind. Dazu maskiert er die Komponenten des Featurevektors mit gleichverteilten Zufallswerten r_i, um sie vor Alice zu verschleiern:

$$[\![x_i]\!] = [\![\overline{\omega}_i + r_i]\!]$$

Diese maskierten Komponenten sendet er an Alice, welche diese mit ihrem privaten Schlüssel entschlüsselt und quadriert, das Ergebnis $[\![x_i^2]\!]$ wieder verschlüsselt und dann an Bob zurücksendet. Bob berechnet dann den i-ten Summand von S_3 durch:

$$[\![x_i^2]\!] \cdot [\![\overline{\omega}_i]\!]^{(-2r_i)} \cdot [\![-r_i^2]\!] = [\![(\overline{\omega}_i + r_i)^2 - 2r_i\overline{\omega}_i - r_i^2]\!] = [\![\overline{\omega}_i^2]\!]$$

Die Aufsummierung der $[\![\overline{\omega}_i^2]\!]$ und sowie von $[\![S_1]\!],[\![S_2]\!],[\![S_3]\!]$ kann Bob nun im Chiffreraum homomorph durch Multiplikation der Chiffretexte durchführen und erhält so $[\![D(\Omega,\overline{\Omega})]\!]$ (vgl. homomorphe Addition bei Okamoto-Uchiyama 2.4.4).

- *Vergleich* zweier verschlüsselter Zahlen im DGK Kryptosystem. Hier setzten die Autoren das Protokoll von 5.4 ein.

Klassifizierungskriterien:

1. Diese Studie zeigt, dass komplexe Funktionen auf verschlüsselten Daten berechnet werden können ohne auf ein vollhomomorphes Kryptosystem angewiesen zu sein. Es wird eine Optimierung bei der Schlüsselgenerierung verwendet, die spezifisch für Paillier ist. Optimierungen bei der Verschlüsselung durch Vorberechnungen lassen sich auch auf andere semihomomorphe Kryptosysteme übertragen. Um den Eigenface Algorithmus zu beschleunigen, wird das Kryptosystem DGK für den Vergleich von Zahlen eingesetzt.

2. Es wird eine Projektion mit Skalarprodukten und ein quadratischer Abstand zwischen zwei Zahlen berechnet. Diese Berechnungen sind allerdings nur möglich, weil eine der Zahlen unverschlüsselt in die Berechnung eingeht. Weiter wird bei beim quadratischen Abstand auf das Wurzelziehen verzichtet, da man nur an einem relativen Vergleich der Ergebnisse interessiert ist.
3. Bei der Skalarproduktberechnung wäre es Alice möglich durch Erstellen eines Vektors, bestehend aus neutralen Elementen in allen Komponenten, Bobs privaten Vektor u_i zu lesen. Aber im Ablauf des Eigenface Algorithmus erhält Alice das Ergebnis dieser Rechenoperation nicht zurück. Um Zwischenergebnisse des Algorithmus anzugreifen, müssen Alice und Bob vom Ablauf des Eigenface Algorithmus abweichen, was jedoch ausgeschlossen wird, da von einem honest-but-curious Angreifer ausgegangen wird.

5.7 Private predictive analysis on encrypted medical data [BLN14]

Joppe W. Bos et. al stellen ein Verfahren vor, in dem ein statistisches Vorhersagemodell auf komplett verschlüsselten Daten arbeitet. Der Client verschlüsselt die medizinischen Daten mit seinem öffentlichen Schlüssel und übergibt diese dem Server. Dort wird ohne Einsatz des privaten Schlüssels und ohne weitere Interaktion mit dem Clienten eine Analyse auf den verschlüsselten Daten durchgeführt. Der Client erhält im Anschluss das verschlüsselte Ergebnis.

Der Server führt eine logistische Regression auf den homomorph verschlüsselten Daten durch, um die Wahrscheinlichkeit einer Herzkreislauferkrankung zu vorherzusagen. Zum Einsatz kommt ein beschränkt (hier: leveled) vollhomomorphes Kryptosystem [BLLN13] mit Optimierungen aus [Bra] (im Folgenden BLLN genannt). Das BLLN Kryptosystem ist IND-CPA sicher. Die Autoren haben sich für dieses Kryptosystem entschieden, da die Chiffretexte bei homomorphen Multiplikationen nicht expandieren[14], dafür ist gegenüber dem Kryptosystem von Brakerski el al. (5.2) die Multiplikation langsamer.

In dem Vorhersagemodell mit logistischer Regression wird folgende Prädiktionsformel berechnet:

$$P(x) = \frac{e^x}{e^x - 1}.$$

Die Eingabe x ist eine Linearkombination von patientspezifischen Parametern, die mit Regressionskoeffizienten gewichtet sind. Somit ist x homomorph berechenbar. Die Prädiktionsformel wird homomorph berechenbar durch Annäherung mit einer Taylorreihe siebten Grades.

Eine Besonderheit des BLLN Kryptosystems ist die Struktur vom Klartext- und Chiffreraum. Sowohl Klartext als auch Chiffretexte repräsentieren Polynome mit ganzzahligen Koeffizienten der Art $\sum_{i=0}^{n-1} a_i X^i, a_i \in \mathbb{Z}$. Die Elemente sind aus dem Polynomring $R = \mathbb{Z}/(X^n + 1)$. Operationen im Chiffreraum entsprechen der Addition und Multiplikation von Polynomen mod $X^n + 1$.

Anmerkungen: Die Autoren sprechen nicht ausdrücklich von einem honest-but-curious Angreifer. Bei der Sicherheit der Implementierung werden nur kryptografische Angreifer betrachtet. Man geht jedoch davon aus, dass Client und Server nicht vom Protokoll abweichen. Dies entspricht dem honest-but-curious Angreifermodell.

Klassifizierungskriterien:

[14] Es wird weniger Speicherplatz benötigt, dafür ist die Rechenzeit in der Ausführung größer. Wie in 2.4.2 erwähnt ist die Expansion der Chiffretext ein Problem bei Vollhomomorphen Kryptosystemen.

1. Es wird das beschränkt vollhomomorphe Kryptosystem BLLN eingesetzt um Polynome zu berechnen. BLLN wird gegen über BGV präferiert, da Chiffretext bei Multiplikation nicht größer werden.
2. Es wird eine Exponentialfunktion linearisiert, um das Taylorpolynom mit BLLN berechenbar zu machen.
3. Das Kryptosystem ist IND-CPA sicher. Die Autoren gehen nicht auf eine unautorisierte Verformbarkeit der Chiffretexte ein. Der Cloudservice interagiert nicht mehr mit dem Clienten nach Erhalt der verschlüsselten Daten. Um eine unautorisierte Verformbarkeit der Chiffretexte auszunutzen, müsste man von einem Angreifer ausgehen, der neben dem Cloudservice auch den Clienten kontrolliert. Aber in diesem Fall hätte der Angreifer Zugriff auf den privaten Schlüssel und alle hinterlegten Daten.

5.8 Klassifizierung

Im Folgenden sind die Ergebnisse der Untersuchungen aus diesem Kapitel zusammengefasst.

5.8.1 Anwendungsfälle der Kryptosysteme

Die erste Tabelle fasst die Anwendungsfälle der Kryptosysteme zusammen und ist daher nach den Veröffentlichungen sortiert. Neben dem verwendeten Kryptosystem wird angegeben auf welche Kryptobibliothek zurückgegriffen wurde, bzw. ob sich die Autoren sich für eine eigene Implementierung entschieden haben. Da asymmetrische homomorphe Kryptosysteme in Protokollen mit mehreren Parteien eingesetzt werden, wird in der Spalte → *Angreifer* angegeben, ob ein Angreifermodell nach Abschnitt 3.5 betrachtet wird. In der Spalte → *Integritätsprüfung der Chiffretexte* wird angegeben, ob die Autoren die Integrität der Chiffretexte prüfen. Die Spalte → *Chiffretextwechsel* gibt an, ob zwischen mehreren Kryptosystemen in einem Anwendungsfall gewechselt wird.

Anmerkungen zur ersten Tabelle: In den Veröffentlichungen 5.5 und 5.7 interagieren die Parteien nur einmal. Im honest-but-curious Angreifermodell bekommt der Angreifer nicht wieder Zugriff auf den verformten Chiffretext, ohne vom Protokoll abweichen zu müssen. Die einzige andere Möglichkeit für den Angreifer, um an den Chiffretext zu kommen, ist, wenn er beide Parteien kontrolliert. Denn würde der Angreifer beide Parteien kontrollieren, wäre er im Besitz des privaten Schlüssels und könnte alle Chiffretexte entschlüsseln. Bei einem honest-but-curious Angreifermodell geht man jedoch davon aus, dass der Angreifer höchstens $n/2$ der Parteien kontrolliert (hier: 1) (Siehe dazu die Definition in Abschnitt 3.5). In vielen der vorgestellten Anwendungsfälle ist eine unautorisierte Verformbarkeit der Chiffretexte damit *nicht ausnutzbar*.

Veröffentlichung	Kryptosystem (Implementierung)	Angreifer	Integritätsprüfung der Chiffretexte
5.1, [TSCS13]	ElGamal (libcrypt [lib]), Paillier (CryptDB [PRZB11])	Bösartig[15]	Nein, ausdrücklich nicht berücksichtigt
5.2, [BPTG15]	Goldwasser-Micali, Paillier (jeweils eigene Implementierung [RB] in C++ mit GMP [gmp] und NTL [nlt]), BGV (HElib [HEl][HS13])	HBC	Nein
5.3, [NIW+13]	Hash-ElGamal (eigene Implementierung[16])	Bösartig	Signierung mit MAC-Code
5.4, [DGK07]	DGK (eigene Implementierung in Java 1.5 unter Verwendung der Klasse BigInteger)	HBC	Nein
5.5, [KT05]	Okamoto-Uchiyama (eigene Implementierung)	HBC	unautorisierte Verformbarkeit nicht ausnutzbar: ein verformter Chiffretext wird im Protokoll nicht mehr zurück erhalten und der Angreifer kontrolliert höchstens eine Partei
5.6, [EFG+09]	Paillier, DGK (jeweils eigene Implementierung mit kryptografischen Optimierungen aus [DJ01] in C++ mit GMP [gmp])	HBC	Nicht ausnutzbar im HBC Modell, da nicht vom Protokoll abgewichen wird.
5.7, [BLN14]	BLLN (eigene Implementierung mit kryptografischen Optimierungen aus [Bra])	HBC	unautorisierte Verformbarkeit nicht ausnutzbar: ein verformter Chiffretext wird im Protokoll nicht mehr zurück erhalten und der Angreifer kontrolliert höchstens eine Partei

Veröffentlichung	Chiffretextwechsel	Grund für Chiffretextwechsel
5.1, [TSCS13]	ElGamal ↔ Paillier	Durchführung von Addition und Multiplikation mit gleicher Variable
5.2, [BPTG15]	Goldwasser-Micali → BGV Goldwasser-Micali → Paillier	Evaluierung von Polynomen, Weiterrechnen unter Paillier
5.3, [NIW+13]	-	-
5.4, [DGK07]	-	-
5.5, [KT05]	-	-
5.6, [EFG+09]	Paillier → DGK	schnellere Vergleiche
5.7, [BLN14]	-	-

[15] Das transformierte Programm läuft auf einem nicht vertrauenswürdigen Webserver. Der Angreifer kann die Eingaben des Programms ersetzen. Dies entspricht dem Modell eines bösartigen Angreifers.

[16] Die Autoren haben die Algorithmen des Kryptosystems vermutlich selber implementiert. Sie machen hierzu jedoch keine Aussage.

5.8.2 Auswahlkriterien und Eigenschaften der Kryptosysteme

Die zweite Tabelle fasst Auswahlkriterien der homomorphen Kryptosysteme zusammen, die in den Veröffentlichungen genannt wurden. Dazu werden grundlegende Eigenschaften der Kryptosysteme aufgelistet.

→ *LVHE* steht für leveled vollhomomorphes Kryptosystem, → *SH* steht für semihomomorphes Kryptosystem, → *OP* steht bei semihomomorphen Kryptosystemen für realisierte Rechenoperationen mit diesem Kryptosystem, → *konst-** steht für Konstantenmultiplikation durch Potenzierung.

Kryptosystem	Auswahlkriterien	Typ	OP	Sicherheit
BGV [BGV12]	Evaluation von Polynomen	LVHE	LVHE	IND-CPA
BLLN [BLLN13]	Evaluation von Polynomen, keine Chiffretextexpansion bei Multiplikation, jedoch langsamer als BGV	LVHE	LVHE	IND-CPA
DGK [DGK07]	Speziell entwickelt für schnelle Vergleiche, kleiner Klartextraum	SH	+	IND-CPA
ElGamal [YPB14]	Homomorphe Multiplikation	SH	*	IND-CCA1
Goldwasser-Micali [GM82]	Vergleiche nach Protokoll [Veu11]	SH	XOR	IND-CPA
Hash-ElGamal [NIW+13]	Effizienter als additive SH Kryptosysteme um Chiffretexte zu maskieren	SH	XOR	IND-CPA
Okamoto-Uchiyama [OU]	Einfachere Implementierung als bei Paillier, da weniger Rechenschritte	SH	+, konst-*	IND-CPA
Paillier [Pai99]	Großer Klartextraum ermöglicht Verschlüsselung von Fließkommazahlen durch Multiplikation mit großen Exponenten, XOR durch Verschlüsselung der Integerbits in 5.1, privates Skalarprodukt, Vergleiche nach Protokoll [Veu11]	SH	+, XOR, $\langle\cdot,\cdot\rangle$, konst-*	IND-CPA

Damit lassen sich folgende Klassen nach Einsatzszenarien identifizieren:

1. *Evaluation von Polynomen* durch leveled vollhomomorphe Kryptosysteme: BGB, BLLN. Obwohl sich Polynome durch Kombinationen von semihomomorphen Kryptosystemen berechnen lassen, wurden leveled vollhomomorphe Kryptosysteme eingesetzt.
2. *Vergleiche* durch spezialisierte Kryptosysteme: DGK. Das Kryptosystem ist für eine schnelle Ausführung optimiert.
3. *Flexibel einsetzbare* Kryptosysteme: Paillier, Okamoto-Uchiyama. Beide Kryptosysteme verschlüsseln Ganzzahlen, können aber auch auf der Bitdarstellung der Ganzzahlen verwendet werden, um Bitoperationen wie XOR im Chiffreraum zu ermöglichen. Beide Kryptosysteme ermöglichen eine Konstantenmultiplikationen durch Potenzierung, wodurch komplexere Operatoren berechnet werden können (Skalarprodukt, quadratischer Abstand).

Weiter gibt es Kryptosysteme, in denen die Verschlüsselungsfunktion teilweise im Voraus berechenbar ist. Da diese Klasse jedoch kein Einsatzszenario festlegt, wird diese Klasse gesondert angegeben:

(4.) *Vorausberechenbare* Kryptosysteme: Paillier, Okamoto-Uchiyama[17], DGK. Faktoren der Verschlüsselungsfunktion, die nicht vom Klartext abhängen, können im Voraus werden. Dazu müssen zu Zufallszahlen erzeugt werden.

[17] In Abschnitt 2.4.4 kann der Faktor h^r vorausberechnet werden.

6 Verwandte Arbeiten

Im Folgenden wird werden mehrere verwandte Arbeiten kurz vorgestellt.

In [MPS12] werden verschiedene semi- und vollhomorphe Kryptosysteme als generelle Lösung für Vertraulichkeitsprobleme im Cloud Computing genannt. Sie erwarten, dass homomorphe Kryptographie in der Zukunft effizienter sein wird und somit ihre Relevanz für den Praxiseinsatz steigt. In Abschnitt 3 werden erste Beispiele für einen Einsatz in Cloud Computing, sicheren Wahlsystemen und privaten Informationsrückgewinnungssystemen erwähnt. Der Einsatz von homomorpher Kryptographie geschieht nur unter dem Hintergrund der Sicherung von Vertraulichkeit. Das Ausnutzen von einer Verformbarkeit der Chiffretexte durch einen Angreifer ist nicht Untersuchungsgegenstand.

Der Artikel [FG07] dient als Übersicht und Einführung in homomorphe Kryptographie für Wissenschaftler, die mit Kryptographie nicht vertraut sind. Die Autoren betonen ausdrücklich die Verformbarkeit von Chiffretexten in homomorphen Kryptosystemen und stellen klar, dass diese geforderte Eigenschaft das Erreichen einer Sicherheit gegen stärkere Angriffsmodelle in der Kryptoanalyse unmöglich macht. Sie gehen darauf ein, dass die Forderung nach semantischer Sicherheit (3.2.2) zur Folge hat, dass eingesetzte Kryptosysteme probabilistisch sein müssen. Das Ausnutzen von einer Verformbarkeit der Chiffretexte durch einen Angreifer wird auch hier nicht untersucht.

Beide Quellen nennen vollhomomorphe Kryptographie unpraktikabel für den praktischen Einsatz wegen hoher Anforderung an verfügbare Ressourcen.

Als relativ junger Teilbereich der Kryptographie lassen sich in den Veröffentlichungen verschiedene Definitionen zu vollhomomorphen Kryptosystemen finden. In [ABC+15] werden diese unter einem rein theoretischen Hintergrund systematisch gegliedert und Bezüge der verschiedenen Definitionen zueinander hergestellt.

7 Zusammenfassung

In dieser Arbeit wurden verschiedene Anwendungsfälle von homomorphen Kryptosystemen untersucht. Dabei wurden die Gründe für die Auswahl des eingesetzten Kryptosystems identifiziert.

In homomorphen Kryptosystemen können erzeugte Chiffretexte alleine durch Kenntnis des öffentlichen Schlüssels von unautorisierten Dritten verformt werden. Mit den Grundlagen aus den Kapiteln 2 und 3 wurde dann in Kapitel 4 formal gezeigt, dass alle semihomomorphen Kryptosystem nicht NM-Sicher sind für die Angreifermodelle CPA, CCA1 und CCA2. Anschließend wurden in Kapitel 5 Anwendungsfälle von homomorphen Kryptosystemen untersucht. Dabei wurde ausgewertet, wie mit möglichen Integritätsverletzungen durch eine unautorisierte Verformung der Chiffretexte umgegangen wird.

Die Untersuchung der Anwendungsfälle ergab, dass eine unautorisierte Verformung der Chiffretexte beim Einsatz von homomorphen Kryptosysteme oft nicht berücksichtigt werden musste. Aufgrund der geringen Interaktion zwischen den Parteien müssten diese vom Protokoll abweichen können, um mittels unautorisierter Verformbarkeit Einsicht in Chiffretexte erhalten zu können. Dieses wurde jedoch ausgeschlossen, da die Autoren ein honest-but-curious Angreifermodell betrachteten. Nur in einem Fall (5.3) wurde die Integrität der Chiffretexte sichergestellt. Hier wurden die Chiffretexte mit MAC-Codes signiert, bevor sie im Protokoll weiter verarbeitet wurden. Ein Schaltkreis der die Chiffretexte verarbeitet, überprüft den zugehörigen MAC-Code.

Die Klassifizierung der Kryptosysteme ergab eine Präferenz von leveled vollhomomorphen Kryptosystemen um Polynome zu berechnen. Weiter werden die Kryptosysteme von Paillier und Okamoto-Uchiyama flexibel eingesetzt. Beide Kryptosysteme sind additiv-homomorph. Durch geschickte Konstruktion sind jedoch auch Multiplikationen, binäre Operationen oder komplexere Verknüpfungen wie das Skalarprodukt mit ihnen berechenbar. Es wurde mit DGK ein spezialisiertes Kryptosystem identifiziert, das speziell entworfen wurde um verschlüsselte Zahlen schnell vergleichen zu können. Viele Anwendungsfälle setzen homomorphe Kryptosysteme ein, um bekannte Protokolle oder Algorithmen auf verschlüsselten Daten zu realisieren, und so eine Privatsphäre wahrende Datenverarbeitung zu gewährleisten (5.1, 5.2, 5.3, 5.6).

Folgende wiederkehrende Herangehensweisen lassen sich bei der Implementierung von Kryptosystemen ausfindig machen:

- *Linearisierung* von Rechnungen: In 5.6 wurde anstelle des euklidischen Abstands nur der quadratische Abstand betrachtet, um die Funktion mit semihomomorphen Kryptosystemen berechenbar zu machen. In 5.7 wurde durch eine Linearisierung mit Taylor die Exponentialfunktion angenähert.
- *Beschleunigungen* von Verschlüsselungsalgorithmen durch Berechnung von Faktoren, die nicht direkt von Klartext abhängen.

7.1 Ausblick

Die Klassifizierung ergab, dass eine unautorisierte Verformbarkeit der Chiffretexte nicht im betrachteten honest-but-curious Angreifermodell ausgenutzt werden konnte. Dieses Angreifermodell ist realistisch, wenn homomorphe Kryptosysteme eingesetzt werden um Rechenoperationen auf einen Server auszulagern. Der Server tritt dann als Dienstleister auf, der an einer korrekten Ausführung des Protokolls interessiert ist, um sein Geschäftsmodell nicht zu gefährden. Wenn jedoch ein Angreifer diesen Server von dem Dienstleister übernehmen kann, dann muss er sich nicht an das Protokoll halten. Das letztere Szenario wird als eine realistischere Abbildung gehalten und daher sollten weitere Veröffentlichungen untersucht werden, die ausschließlich bösartige Angreifer betrachten.

Das Ziel einer Sicherung der Integrität von homomorphen Rechenoperationen überschneidet sich mit dem Begriff des *Verifiable Computing*. Verifiable Computing betrachtet einen „schwachen“ Clienten, der Rechenoperationen an einen oder mehrere „Arbeiter“ auslagert. Mit Verifiable Computing möchte man ein Protokoll bereitstellen in dem die Arbeiter 1) das Ergebnis einer gewünschten Berechnung des Clienten zurückgeben 2) sowie einen Beweis, dass diese Berechnung korrekt ausgeführt wurde.

Verifiable Computing wurde 2010 von Gennaro, Gentry und Parno formalisiert [GGP10]. Darauf basierte Protokolle könnten eine Lösung für die unautorisierte Verformbarkeit bei homomorphen Kryptosystem bieten.

8 Literaturverzeichnis

[ABC+15] Armknecht, Frederik ; Boyd, Colin ; Carr, Christopher ; Gjøsteen, Kristian ; Jäschke, Angela ; Reuter, Christian A. ; Strand, Martin: A Guide to Fully Homomorphic Encryption. In: *IACR Cryptology ePrint Archive* (2015), S. 1192

[Ado06] Adobe® Systems Incorporated: *PDF Reference, Adobe® Portable Document Format, Version 1.7*. http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/pdf_reference_1-7.pdf, 2006. – (Aufgerufen am 12. Mai 2017)

[AKP13] Armknecht, Frederik ; Katzenbeisser, Stefan ; Peter, Andreas: Group homomorphic encryption: characterizations, impossibility results, and applications. In: *Designs, codes and cryptography* (2013), S. 1–24

[BCC88] Brassard, Gilles ; Chaum, David ; Crépeau, Claude: Minimum disclosure proofs of knowledge. In: *Journal of Computer and System Sciences* 37 (1988), Nr. 2, S. 156–189

[BDPR] Bellare, Mihir ; Desai, Anand ; Pointcheval, David ; Rogaway, Phillip: Relations among notions of security for public-key encryption schemes. In: *Advances in Cryptology—CRYPTO'98* Springer, S. 26–45

[BGN05] Boneh, Dan ; Goh, Eu-Jin ; Nissim, Kobbi: Evaluating 2-DNF formulas on ciphertexts. In: *Theory of Cryptography Conference* Springer, 2005, S. 325–341

[BGV12] Brakerski, Zvika ; Gentry, Craig ; Vaikuntanathan, Vinod: (Leveled) fully homomorphic encryption without bootstrapping. In: *ITCS*, 2012

[BL96] Boneh, Dan ; Lipton, Richard J.: Algorithms for black-box fields and their application to cryptography. In: *Annual International Cryptology Conference* Springer, 1996, S. 283–297

[Ble] Bleichenbacher, Daniel: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1. In: *Advances in Cryptology—CRYPTO'98* Springer, S. 1–12

[BLLN13] Bos, Joppe W. ; Lauter, Kristin E. ; Loftus, Jake ; Naehrig, Michael: Improved Security for a Ring-Based Fully Homomorphic Encryption Scheme. In: *IMA Int. Conf.* Springer, 2013, S. 45–64

[BLN14] Bos, Joppe W. ; Lauter, Kristin ; Naehrig, Michael: Private predictive analysis on encrypted medical data. In: *Journal of biomedical informatics* 50 (2014), S. 234–243

[BPTG15] Bost, Raphael ; Popa, Raluca A. ; Tu, Stephen ; Goldwasser, Shafi: Machine Learning Classification over Encrypted Data. In: *NDSS*, 2015

[Bra] Brakerski, Zvika: Fully homomorphic encryption without modulus switching from classical GapSVP. In: *Advances in Cryptology–CRYPTO 2012*. Springer, S. 868–886

[BS99] Bellare, Mihir ; Sahai, Amit: Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. In: *Annual International Cryptology Conference* Springer, 1999, S. 519–536

[BSW95] Beutelspacher, Albrecht ; Schwenk, Jörg ; Wolfenstetter, Klaus-Dieter: *Moderne Verfahren der Kryptographie*. Vieweg, 1995

[Buc04] Buchmann, Johannes: *Introduction to cryptography*. Springer Science & Business Media, 2004

[CMPP06] Chevallier-Mames, Benoît ; Paillier, Pascal ; Pointcheval, David: Encoding-free ElGamal encryption without random oracles. In: *International Workshop on Public Key Cryptography* Springer, 2006

[Cry] *Cryptosystem - Wikipedia*. https://en.wikipedia.org/wiki/Cryptosystem, . – (Aufgerufen am 28. März 2017)

[DDN03] Dolev, Danny ; Dwork, Cynthia ; Naor, Moni: Nonmalleable cryptography. In: *SIAM review* 45 (2003), Nr. 4, S. 727–784

[DGK07] Damgård, Ivan ; Geisler, Martin ; Krøigaard, Mikkel: Efficient and secure comparison for on-line auctions. In: *Australasian Conference on Information Security and Privacy* Springer, 2007, S. 416–430

[DH76] Diffie, Whitfield ; Hellman, Martin: New directions in cryptography. In: *IEEE transactions on Information Theory* 22 (1976), Nr. 6, S. 644–654

[DJ01] Damgård, Ivan ; Jurik, Mads: A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system. In: *International Workshop on Public Key Cryptography* Springer, 2001, S. 119–136

[DKK02] Delfs, Hans ; Knebl, Helmut ; Knebl, Helmut: *Introduction to cryptography*. Bd. 2. Springer, 2002

[EFG+09] Erkin, Zekeriya ; Franz, Martin ; Guajardo, Jorge ; Katzenbeisser, Stefan ; Lagendijk, Inald ; Toft, Tomas: Privacy-preserving face recognition. In: *International Symposium on Privacy Enhancing Technologies Symposium* Springer, 2009, S. 235–253

[FG07] Fontaine, Caroline ; Galand, Fabien: A survey of homomorphic encryption for nonspecialists. In: *EURASIP Journal on Information Security* (2007), Nr. 1, S. 1–10

[Fis10] Fischer, Gerd: *Lineare Algebra, Vieweg+ Teubner, Wiesbaden, 17.* 2010

[Gen09] Gentry, Craig: *A fully homomorphic encryption scheme*, Stanford University, Diss., 2009

[GGP10] Gennaro, Rosario ; Gentry, Craig ; Parno, Bryan: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: *Annual Cryptology Conference* Springer, 2010, S. 465–482

[GM82] Goldwasser, Shafi ; Micali, Silvio: Probabilistic encryption & how to play mental poker keeping secret all partial information. In: *Proceedings of the fourteenth annual ACM symposium on Theory of computing* ACM, 1982, S. 365–377

[gmp] *GNU Multiple Precision Library (GMP)*. https://gmplib.org/, . – (Aufgerufen am 21. Mai 2017)

[Gol93] Goldreich, Oded: A uniform-complexity treatment of encryption and zero-knowledge. In: *Journal of Cryptology* 6 (1993), Nr. 1, S. 21–53

[HEl] *HELib*. https://github.com/shaih/HElib, . – (Aufgerufen am 21. Mai 2017)

[Hof10] Hoffmann, Dirk W.: *Grundlagen der technischen Informatik*. Carl Hanser Verlag GmbH Co KG, 2010

[HS13] Halevi, Shai ; Shoup, Victor: Design and implementation of a homomorphic-encryption library. In: *IBM Research (Manuscript)* 6 (2013), S. 12–15

[KBV09] Koren, Yehuda ; Bell, Robert ; Volinsky, Chris: Matrix factorization techniques for recommender systems. In: *Computer* 42 (2009), Nr. 8

[KL14] Katz, Jonathan ; Lindell, Yehuda: *Introduction to modern cryptography*. CRC press, 2014

[KT05] Kuribayashi, Minoru ; Tanaka, Hatsukazu: Fingerprinting protocol for images based on additive homomorphic property. In: *IEEE Transactions on Image Processing* 14 (2005), Nr. 12, S. 2129–2139

[lib] *HELib*. https://gnupg.org/related_software/libgcrypt/, . – (Aufgerufen am 21. Mai 2017)

[MPS12] Maimut, Diana S. ; Patrascu, Alecsandru ; Simion, Emil: Homomorphic encryption schemes and applications for a secure digital world. In: *Journal of Mobile, Embedded and Distributed Systems* 4 (2012), Nr. 4, S. 224–232

[MVOV96] Menezes, Alfred J. ; Van Oorschot, Paul C. ; Vanstone, Scott A.: *Handbook of applied cryptography*. CRC press, 1996

[NIW+13] Nikolaenko, Valeria ; Ioannidis, Stratis ; Weinsberg, Udi ; Joye, Marc ; Taft, Nina ; Boneh, Dan: Privacy-preserving matrix factorization. In: *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* ACM, 2013, S. 801–812

[nlt] *NTL: A Library for doing Number Theory*. http://www.shoup.net/ntl/, . – (Aufgerufen am 21. Mai 2017)

[NS98] Naccache, David ; Stern, Jacques: A new public key cryptosystem based on higher residues. In: *Proceedings of the 5th ACM conference on Computer and communications security* ACM, 1998, S. 59–66

[OU] Okamoto, Tatsuaki ; Uchiyama, Shigenori: A new public-key cryptosystem as secure as factoring. In: *Advances in Cryptology—EUROCRYPT'98*

[Pai99] Paillier, Pascal: Public-key cryptosystems based on composite degree residuosity classes. In: *International Conference on the Theory and Applications of Cryptographic Techniques* Springer, 1999, S. 223–238

[PRZB11] Popa, Raluca A. ; Redfield, Catherine ; Zeldovich, Nickolai ; Balakrishnan, Hari: CryptDB: protecting confidentiality with encrypted query processing. In: *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles* ACM, 2011, S. 85–100

[RB] Raphael Bost, Stephen T. Raluca Ada Popa P. Raluca Ada Popa: *Implementation of "Machine Learning Classification over Encrypted Data"*. https://github.com/rbost/ciphermed/tree/master/src/crypto/, . – (Aufgerufen am 21. Mai 2017)

[RSA78] Rivest, Ronald L. ; Shamir, Adi ; Adleman, Leonard: A method for obtaining digital signatures and public-key cryptosystems. In: *Communications of the ACM* 21 (1978), Nr. 2, S. 120–126

[Sch11] Schneier, Bruce: *Secrets and lies: digital security in a networked world*. John Wiley & Sons, 2011

[Sha49] Shannon, Claude E.: Communication theory of secrecy systems. In: *Bell Labs Technical Journal* 28 (1949), Nr. 4, S. 656–715

[Sma03] Smart, Nigel P.: *Cryptography: An Introduction*. Bd. 5. McGraw-Hill New York, 2003

[Sti06] Stinson, Douglas R.: *Cryptography: theory and practice*. CRC press, 2006

[TSCS13] Tople, Shruti ; Shinde, Shweta ; Chen, Zhaofeng ; Saxena, Prateek: AUTOCRYPT: enabling homomorphic computation on servers to protect sensitive web content. In: *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* ACM, 2013, S. 1297–1310

[Veu11] Veugen, Thijs: Comparing encrypted data. In: *Multimedia Signal Processing Group, Delft University of Technology, The Netherlands, and TNO Information and Communication Technology, Delft, The Netherlands, Tech. Rep* (2011)

[WS08] Wu, Jiang ; Stinson, Douglas R.: On The Security of The ElGamal Encryption Scheme and Damgard's Variant. In: *IACR Cryptology ePrint Archive* (2008), S. 200

[Yao82] Yao, Andrew C.: Protocols for secure computations. In: *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on* IEEE, 1982, S. 160–164

[YPB14] Yi, Xun ; Paulet, Russell ; Bertino, Elisa: *Homomorphic encryption and applications*. Bd. 3. Springer, 2014