Hochschule Wismar
University of Applied Sciences
Technology, Business and Design
Fakultät für Ingenieurwissenschaften, Bereich EuI

Master-Thesis

Möglichkeiten und Grenzen Isogeniebasierter Schlüsselaustauschverfahren

Capabilities and Limits of Isogeny-Based Key-Exchange Algorithms

Eingereicht am: 23. September 2021

von: Marian Franke

Aufgabenstellung

Zielsetzung

Das Ziel dieser Masterarbeit ist es, Schlüsselaustauschverfahren in der Post Quantum Cryptography (PQC) zu analysieren und ihre Nutzbarkeit anhand fester Kriterien zu evaluieren. Der Fokus liegt bei der Betrachtung auf Verfahren, die einen Schlüsselaustausch auf Basis von elliptischen Kurven und Isogenien realisieren. Dabei sollen die Vor- und Nachteile verschiedener isogeniebasierter Verfahren mit aktuell zur Anwendung kommenden Verfahren gegenübergestellt sowie Angriffsvektoren identifiziert werden. Es werden Komplexitätsüberlegungen zur Schlüsselerzeugung und zur Ver- und Entschlüsselung in Betracht gezogen, sowie Aufwandsüberlegungen anhand von Laufzeit- und Speicherbedarf angestellt. Die Risiken verschiedener Verfahren werden nach gängigen Risk-Assessment-Modellen eingeschätzt. Hierfür sollen Implementierungen der betrachteten Algorithmen nachvollzogen und in einem Test-Setup analysiert werden.

Abgrenzung

Im Rahmen dieser Masterarbeit werden keine neuen Verfahren zum Schlüsselaustausch in der PQC entwickelt. Die betrachteten Verfahren werden nicht vollständig umgesetzt, stattdessen werden einzelne Aspekte simulativ betrachtet. Hierbei wird sich an den Verfahren orientiert, die vom National Institute of Standards and Technology (NIST) als Kandidaten in ihren Bestrebungen zur Standardisierung der PQC betrachtet werden. Es können nicht alle Angriffsvektoren auf die betrachteten Verfahren berücksichtigt werden, da die abschließende Sicherheitsbetrachtung einen Quantencomputer benötigen würde.

Inhaltsverzeichnis

1 Einleitung **5**

2 Mathematische Grundlagen **8**
2.1 Elliptische Kurven . . . 8
2.2 j-Invariante . . . 10
2.3 Isogenien . . . 10
2.4 Velu's Formeln . . . 11

3 Kryptographische Grundlagen **13**
3.1 Klassische Kryptographie . . . 13
3.1.1 Symmetrische Kryptographie . . . 13
3.1.2 Asymmetrische Kryptographie . . . 14
3.2 Post-Quanten-Kryptographie . . . 15
3.2.1 Multivariate Verfahren . . . 15
3.2.2 Codebasierte Verfahren . . . 16
3.2.3 Hashbasierte Verfahren . . . 16
3.2.4 Gitterbasierte Verfahren . . . 16
3.2.5 Isogeniebasierte Verfahren . . . 17
3.2.6 Vergleich . . . 17

4 Schlüsselaustauschverfahren **18**
4.1 Klassischer Diffie-Hellman . . . 18
4.2 Diffie-Hellmann mit Elliptischen Kurven . . . 19
4.3 Isogeniebasierter Schlüsselaustausch . . . 21

5 Tests der Schlüsselaustauschverfahren **25**
5.1 Allgemeines Setup . . . 25
5.2 Diffie-Hellman . . . 26
5.3 Elliptic Curve Diffie-Hellman . . . 27
5.4 Supersingular Isogeny Diffie-Hellmann (SIDH) . . . 28
5.5 Commutative Supersingular Isogeny Diffie-Hellmann (CSIDH) . . . 29
5.6 Vergleich . . . 30

6 Risikoanalyse der Schlüsselaustauschverfahren **32**
6.1 Allgemeines . . . 32
6.2 Verfahren zur Risikoanalyse . . . 32
6.2.1 DREAD . . . 33
6.2.2 STRIDE . . . 33
6.2.3 PASTA . . . 34
6.2.4 Vergleich der Risikoanalyseverfahren . . . 35

6.3 Diffie-Hellman . . . 37
6.3.1 Wahl der Primzahl p . . . 38
6.3.2 Wahl der Basis g . . . 38
6.3.3 Vorausberechenbarkeit . . . 39
6.3.4 Man-in-the-Middle . . . 40
6.3.5 Der Diskrete Logarithmus auf einem Quantencomputer . . . 41
6.3.6 Zusammenfassung . . . 42
6.4 Elliptic Curve Diffie-Hellman . . . 43
6.4.1 Wahl der Parameter . . . 43
6.4.2 Schlüsselvalidierung . . . 44
6.4.3 Man-in-the-Middle . . . 46
6.4.4 Der Diskrete Logarithmus auf einem Quantencomputer . . . 46
6.4.5 Zusammenfassung . . . 47
6.5 Supersingular Isogeny Diffie-Hellmann (SIDH) . . . 48
6.5.1 Wahl der Primzahl p . . . 48
6.5.2 Wiederverwendung von Schlüsseln . . . 49
6.5.3 Meet-in-the-Middle . . . 50
6.5.4 Zusammenfassung . . . 51
6.6 Commutative Supersingular Isogeny Diffie-Hellmann (CSIDH) . . . 52
6.6.1 Wahl der Primzahl p . . . 52
6.6.2 Hidden-shift-Problem . . . 52
6.6.3 Meet-in-the-Middle . . . 53
6.6.4 Zusammenfassung . . . 54
6.7 Vergleich . . . 54

7 Fazit **56**
7.1 Zusammenfassung . . . 56
7.2 Ausblick . . . 57
7.2.1 Informationsabfluss durch veröffentlichte Punkte in SIDH . . . 57
7.2.2 Laufzeit von CSIDH . . . 57
7.2.3 Quantum-Sicherheitsanalyse von CSIDH . . . 57

Literaturverzeichnis **59**

Abbildungsverzeichnis **64**

Abkürzungen **65**

Selbstständigkeitserklärung **66**

1 Einleitung

In der Geschichte der Menschheit war die Notwendigkeit zum vertraulichen Austausch von Nachrichten stets gegeben. Also wurden Nachrichten so verschlüsselt, dass niemand anderes als der berechtigte Empfänger diese lesen konnte. Vor der Moderne kannten sich Empfänger und Absender in der Regel bereits vor dem Nachrichtenaustausch und konnten sich so zuvor auf ein gemeinsames Geheimnis einigen, das den Schlüssel zur Entzifferung verschlüsselter Nachrichten darstellte. Bis in die 1970er Jahre bestand Ver- und Entschlüsselung ausschließlich aus sogenannter symmetrischer Kryptographie, bei der die Schlüssel zur Ver- und Entschlüsselung gleich sind. Beide an der Kommunikation beteiligten Parteien einigen sich dabei auf ein gemeinsames Geheimnis, mit welchem Nachrichten ver- und entschlüsselt werden und so von Unbekannten nicht gelesen werden können.

Im modernen Informationszeitalter kam jedoch das Problem auf, dass sich Empfänger und Absender nicht bekannt sind und so keine Möglichkeit hatten, sich zuvor auf einen gemeinsamen Schlüssel zu einigen. Insbesondere im weltweiten Kommunikationsnetz des Internet sollten informationstechnische Geräte untereinander kommunizieren können, ohne dass eine zusätzliche menschliche Interaktion oder ein zusätzlicher Kommunikationskanal notwendig ist. Eine Lösung dieses Problems stellt der Algorithmus nach Diffie Hellman (DH) dar, der es zwei Parteien ermöglicht, sich über einen potenziell unsicheren Gesprächskanal auf ein gemeinsames Geheimnis zu einigen, ohne dass ein eventueller „Mithörer“ dieses Geheimnis erfahren kann. Die Erfindung des DH gilt als Gründung der asymmetrischen Kryptographie, bei welcher die Schlüssel für Ver- und Entschlüsselung einer Nachricht unterschiedlich sind. Dies wird durch eine Public-Key-Infrastruktur gelöst, bei der beide an der Kommunikation beteiligten Parteien den öffentlich verfügbaren Schlüssel der jeweils anderen Partei zur Verschlüsselung der Nachricht verwenden. Die Entschlüsselung dieser Nachricht ist nur mit dem zum öffentlichen Schlüssel zugehörigen privaten Schlüssel möglich, welcher ausschließlich der Partei bekannt ist, für welche die Nachricht bestimmt ist.

Die Sicherheit der heute eingesetzten Verfahren nach DH basiert dabei auf der An-

nahme, dass es durch die heute einsetzbaren Computer zwar sehr einfach ist, Zahlen zu potenzieren, die Umkehrfunktion jedoch sehr rechenaufwändig ist und es somit in absehbarer Zeit nicht möglich ist, aus einer gegebenen Basis und einem bekannten Potenzwert den Exponenten zu bestimmen. Mathematisch spricht man dabei vom Discrete Logarithm Problem (DLP). Mit der Erforschung von Quantencomputern wurde ein Verfahren entdeckt, bei welchem davon auszugehen ist, dass dieses den diskreten Logarithmus in endlicher Zeit lösen kann, und somit den verschlüsselten Nachrichtenaustausch gefährdet. Es wurde festgestellt, dass vom Aufkommen praktisch nutzbarer Quantencomputer sowohl symmetrische als auch asymmetrische Kryptographie bedroht sind. Dabei kann die symmetrische Kryptographie dieser Gefahr jedoch begegnen, indem größere Schlüssellängen verwendet werden. Die assymetrische Kryptographie basiert jedoch auf mathematischen Problemen, die mit einem Quantencomputer ihre Schwierigkeit verlieren und daher insgesamt als unsicher gelten.

Im Rahmen der PQC wird daher daran geforscht, Verfahren zu entwickeln, die den Geheimnisaustausch auch in einem Zeitalter der Quantencomputer sicher bewerkstelligen. Das US-amerikanische NIST hat im Jahr 2016 eine Ausschreibung gestartet, um asymmetrische kryptographische Verfahren zu entwickeln, die eine vertrauliche Kommunikation auch unter quantenkryptographischen Aspekten ermöglichen. Im Rahmen der NIST-Ausschreibung wurden fünf Kategorien quantenkryptographischer Verfahren identifiziert, welche den Möglichkeiten eines Quantencomputers standhalten können. Die Verfahren der PQC werden in die Kategorien der multivariaten, codebasierten, gitterbasierten, hashbasierten und isogeniebasierten Kryptographie unterteilt. Für die NIST-Ausschreibung wurde auch das sogenannte Supersingular Isogeny Diffie Hellman (SIDH) eingereicht, das als isogeniebasiertes Verfahren ein Kandidat ist, um den klassischen DH-Schlüsselaustausch zu ersetzen. Im Rahmen dieser Masterarbeit sollen isogeniebasierte Verfahren analysiert und gegenübergestellt werden.

Nach der Erläuterung der mathematischen und kryptographischen Grundlagen werden in der vorliegenden Arbeit die Funktionsweisen aktueller und isogeniebasierter Schlüsselaustauschverfahren vorgestellt. Im nächsten Abschnitt werden Zeitmessungen der klassischen und isogeniebasierten Verfahren durchgeführt, um aktuelle Implementierungen auf einer klassischen Computerarchitektur miteinander zu vergleichen. Anschließend werden bekannte Schwachstellen und Angriffsvektoren auf klassische und isogeniebasierte Verfahren vorgestellt und erläutert. In einer Risikoanalyse werden diese Angriffsvektoren bewertet und so miteinander verglichen.

Abschließend wird aus diesen gewonnenen Erkenntnissen eine Bewertung der Praktikabilität der Verfahren vorgenommen, um eine Einschätzung vorzunehmen.

2 Mathematische Grundlagen

2.1 Elliptische Kurven

Elliptische Kurven sind Kurven mit speziellen algebraischen Eigenschaften, die sie für viele Anwendungsfelder sowohl in der klassischen Kryptographie als auch in der PQC attraktiv machen. Für die Beschreibung von elliptischen Kurven sind verschiedene Formen bekannt, üblich ist die Darstellung in verkürzter Weierstraß-Form $E:\ y^2 = x^3 + ax + b$.

Auf elliptischen Kurven sind verschiedene Operationen geometrisch definiert. Sind auf einer elliptischen Kurve E die beiden Punkte P und Q gegeben, lassen sich diese beiden Punkte miteinander addieren, indem wie in Abb. 2.1 dargestellt durch die beiden Punkte P und Q eine Gerade gezogen wird, welche die Kurve in einem dritten Punkt schneidet. Dieser Punkt wird an der x-Achse gespiegelt und ergibt so den Punkt $P + Q$.

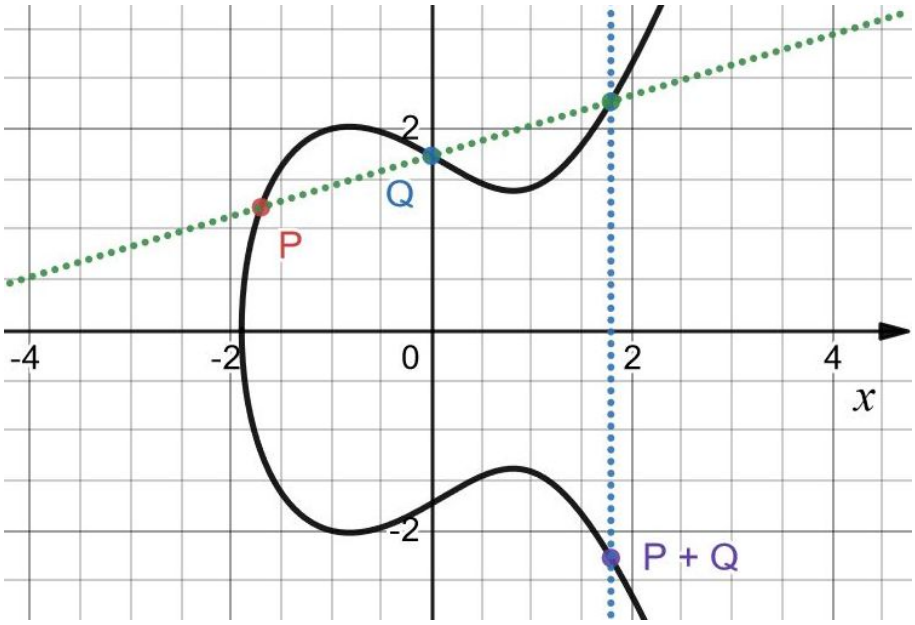

Abb. 2.1: Punktaddition $P + Q$ auf $E : y^2 = x^3 - 2x + 3$

Die Punktmultiplikation $n \cdot P$ wird als n-fache Addition des Punktes P mit sich selbst definiert. Geometrisch wird dies in Abb. 2.2 veranschaulicht. Statt einer Gerade durch zwei Punkte wird dabei die Tangente durch den Punkt P gezogen, welche die Kurve E in einem weiteren Punkt schneidet. Dieser Punkt wird wiederum an der x-Achse gespiegelt, wodurch sich die Punktverdopplung $2 \cdot P$ ergibt. Die mehrfache Durchführung dieser Operation ergibt die skalare Multiplikation $n \cdot P$.

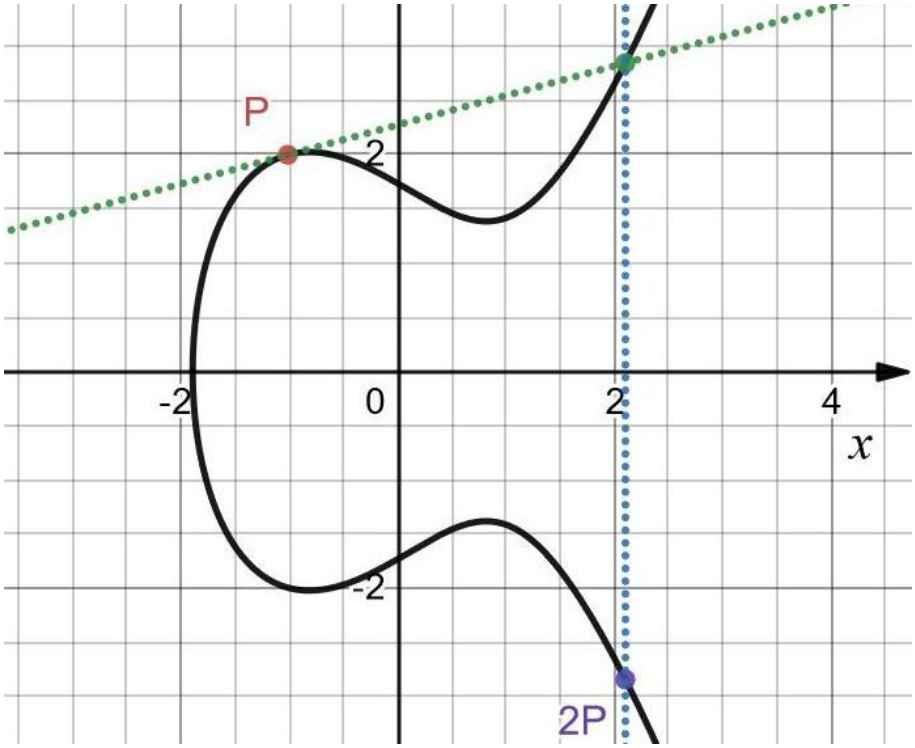

Abb. 2.2: Punktverdopplung $2P$ auf $E : y^2 = x^3 - 2x + 3$

Bezüglich der Addition existiert ein neutrales Element $\mathcal{O}$, sodass $\mathcal{O} + P = P$ für jedes $P \in E$. Das neutrale Element wird auch als *Identitätselement* bezeichnet. Geometrisch gesehen ist das Identitätselement als der *Punkt im Unendlichen* definiert. Werden die Punkte auf E als Koordinaten der projektiven Ebene angegeben, ist das Identitätselement als einziger Punkt der Kurve angegeben, der eine Koordinate $Z = 0$ besitzt, sodass gilt $P = (X : Y : Z)$ mit $Z = 1$ für $P \neq \mathcal{O}$ und $Z = 0$ für $P = \mathcal{O}$. In den Berechnungen werden für $P \neq \mathcal{O}$ häufig nur zwei Koordinaten $P = (X : Y)$ angegeben, wobei die projektive Z-Komponente impliziert wird.

Im Verhältnis zum neutralen Element $\mathcal{O}$ besitzt jeder Punkt P der elliptischen Kurve eine *Ordnung*, die den Abstand des Punktes zum neutralen Element beschreibt. Für jeden Punkt $P \in E$ mit n-ter Ordnung gilt $n \cdot P = \mathcal{O}$. Das neutrale Element $\mathcal{O}$ hat stets die Ordnung 1.

In kryptographischen Anwendungsgebieten wird häufig die sogenannte *Montgomery*-Form $E: \ y^2 = x^3 + Ax^2 + x$ verwendet. Diese ermöglicht gegenüber anderen geläufigen Formen einen verringerten Rechen- und Speicheraufwand, da in der Montgomery-Form elliptische Kurven ausschließlich über die Variable A beschrieben werden können. Außerdem werden Punkte auf der elliptischen Kurve E allein durch ihre X-Koordinate und ein weiteres Bit beschrieben, welches angibt, ob die Y-Koordinate positiv oder negativ ist. Durch die Symmetrie der Kurve kann durch diese Angaben jeder Punkt eindeutig identifiziert werden. Die Montgomery-Form stellt eine elliptische Kurve E in folgender Form dar.

$$E: \ y^2 = x^3 + Ax^2 + x \tag{2.1}$$

2.2 j-Invariante

Die j-Invariante ist spezieller Wert zur Beschreibung von elliptischen Kurven. Jede elliptische Kurve hat eine j-Invariante. Zwei elliptische Kurven E_1 und E_2 gelten als *isomorph* zueinander, wenn beide Kurven die gleiche j-Invariante besitzen $j(E_1) = j(E_2)$.

Zwei Kurven E_1 und E_2 sind zueinander *isomorph*, wenn zwischen ihnen die bijektive Abbildung $\phi: \ E_1 \mapsto E_2$ und das Inverse $\phi^{-1}: \ E_2 \mapsto E_1$ existieren, wobei die Verknüpfung der beiden Abbildungen jeweils das Identitätselement $\mathcal{O}$ ergibt: $\phi \circ \phi^{-1} = \mathcal{O}_{E_2}$ und $\phi^{-1} \circ \phi = \mathcal{O}_{E_1}$. [1]

Zu einer elliptischen Kurve E in Montgomery-Form $E: y^2 = x^3 + Ax^2 + x$ lässt sich die zugehörige j-Invariante nach folgender Formel berechnen.

$$j(E) = \frac{256(A^2 - 3)^3}{(A^2 - 4)} \tag{2.2}$$

2.3 Isogenien

Als Isogenie bezeichnet man eine Abbildung zwischen zwei elliptischen Kurven $\phi: E_1 \mapsto E_2$. Zur Berechnung einer Isogenie wird ein sogenannter Generatorpunkt $G \in E_1$ gewählt, der auf das Identitätselement $\mathcal{O} \in E_2$ abbildet. Der Grad der Isogenie ϕ ergibt sich dabei aus der Ordnung des Erzeugerpunktes G. Aufgrund

der Gruppenoperationen wird ϕ häufig auch als Abbildung von E_1 über der Untergruppe G abgebildet $\phi : E_1 \mapsto E_2 = E_1/\langle G \rangle$. In der Praxis gibt es effiziente Formeln zur Berechnung von Isogenien des Grades 2 und 3 (sogenannte 2-Isogenien bzw. 3-Isogenien), theoretisch können aber Isogenien eines beliebigen Grades n (n-Isogenien) erzeugt werden [2].

Eine Besonderheit von Isogenien ist, dass diese *Homomorphismen* darstellen und somit die Gruppenstruktur der elliptischen Kurven erhalten. Dies bedeutet, dass für die Verknüpfung zweier Punkte $P, Q \in E_1$ gilt $\phi(P \circ Q) = \phi(P) \circ \phi(Q)$.

Punkte höherer Ordnung werden durch Isogenien auf Punkte niedriger Ordnung abgebildet, wenn der Grad der Isogenie die Ordnung des Punktes teilt. Punkte der Ordnung 6 werden also beispielsweise durch eine Isogenie des Grades 2 auf Punkte der Ordnung 3 abgebildet, während die gleichen Punkte durch eine Isogenie des Grades 3 auf Punkte der Ordnung 2 abgebildet werden. Punkte mit zum Grad der Isogenie teilerfremder Ordnung werden auf Punkte gleichbleibender Ordnung abgebildet. Sind die Ordnung des Punktes und der Grad der Isogenie teilerfremd, behalten die Punkte nach der Abbildung ihre Ordnung.

2.4 Velu's Formeln

Auf einer gegebenen elliptischen Kurve $E: \ y^2 = x^3 + Ax^2 + x$ und einem Generatorpunkt $G \in E$ lassen sich mithilfe von Velu's Formeln das Bild der x-Koordinaten und die Zielmenge $E' : y^2 = x^3 + A'x^2 + x$ bestimmen.

Hat der Generatorpunkt $G = (\alpha : 0)$ die Ordnung 2, lassen sich mithilfe von Velu's Formeln Isogenien des Grades 2 wie folgt berechnen [3].

$$A' = 2(1 - 2\alpha^2) \tag{2.3}$$

Es ergibt sich $E \mapsto E'$ mit $E' : \ y^2 = x^3 + A'x^2 + x$.

$$x \mapsto \frac{x(\alpha x - 1)}{x - \alpha} \tag{2.4}$$

Mithilfe von Formel 2.3 lässt sich für die Abbildung $\phi: \ E \mapsto E'$ mit $E': \ y^2 = x^3 + A'x^2 + x$ die Zielmenge bestimmen.

Velu's Formeln liefern ebenso eine Berechnung der Isogenien der Ordnung 3 aus einem Generatorpunkt $G = (\beta : \gamma)$ bzw. $G = (\beta : -\gamma)$ für die Abbildung $\phi: \ E \mapsto E'$ mit $E': \ y^2 = x^3 + A'x^2 + x$.

$$A' = (A\beta - 6\beta^2 + 6)\beta \tag{2.5}$$

$$x \mapsto \frac{x(\beta x - 1)^2}{(x - \beta)^2} \tag{2.6}$$

Die Berechnung mit Velu's Formeln offenbart einen weiteren Vorteil: Da die Abbildung der Punkte der Kurve E ausschließlich abhängig von den x-Werten ist, müssen die y-Werte der Punkte in Folgeberechnungen nicht berücksichtigt werden. Zur weiteren Berechnung ist ausschließlich relevant, ob der Punkt überhalb oder unterhalb der Y-Achse liegt.

3 Kryptographische Grundlagen

3.1 Klassische Kryptographie

Die klassische Kryptographie bezeichnet alle kryptographischen Verfahren, die nicht zur Post-Quanten-Kryptographie zählen. Dies umfasst Verfahren, die eine Ver- und Entschlüsselung ermöglichen, welche auf klassischen Computerarchitekturen als sicher gelten, aber von quantenkryptographischen Verfahren gebrochen werden können bzw. als unsicherer gelten. Üblicherweise unterscheidet man zwischen symmetrischer und asymmetrischer Kryptographie.

3.1.1 Symmetrische Kryptographie

Unter der symmetrischen Kryptographie werden alle kryptographischen Verfahren zusammengefasst, bei denen der Schlüssel zur Verschlüsselung und Entschlüsselung gleich bleibt. Bei zwei an der Kommunikation beteiligten Parteien müssen sich beide zuvor auf ein gemeinsames Geheimnis geeinigt haben. Symmetrische Kryptographie kann auch zur Verschlüsselung von Dateien oder Datenträgern verwendet werden. Hierbei kann man sich vorstellen, dass die „Kommunikation“ zwischen ein und derselben Person stattfindet. Der geheime Schlüssel könnte dabei beispielsweise aus einem Passwort oder einer kryptographischen Datei generiert werden.

Bekannte Verfahren der symmetrischen Kryptographie sind der auf der Feistel-Chiffre basierende Data Encryption Standard (DES) und der auf dem Rijndael-Algorithmus basierende Advanced Encryption Standard (AES). Beide Verfahren funktionieren als Blockchiffren, teilen also eine zu verschlüsselnde Nachricht in Blöcke fester Länge ein, um diese nach einem vorgegebenen Algorithmus zu vertauschen, verschieben und mit einem geheimen Schlüssel zu kombinieren. Grundsatz der Verschlüsselung soll es sein, dass die kleinsten Veränderungen im zu verschlüsselnden Text oder dem geheimen Schlüssel zur möglichst vollständigen Veränderung im resultierenden Geheimtext führen, um beim Vergleich von Geheimtexten keine Informationen über den Ursprungstext oder den geheimen Schlüssel erhalten zu können.

Im Gegensatz zu Blockchiffren existieren auch Stromchiffren, die einen Eingabetext zeichenweise mit einem Schlüsseltext verschlüsselt. Charakteristisch ist bei Stromchiffren, dass aufgrund der Architektur des Schlüsselstroms ein Geheimtext nur am Stück wieder entschlüsselt werden kann. In der Praxis spielen Stromchiffren kaum eine Rolle.

Obwohl symmetrische Kryptographie ebenfalls von der Entwicklung eines Quantencomputers betroffen ist, kann den durch einen Quantencomputer ermöglichten Effizienzsteigerungen in der Kryptoanalyse durch die Wahl von höheren Schlüssellängen entgegengewirkt werden, sodass bisher eingesetzte symmetrische Verschlüsselungsverfahren auch mit der Entwicklung eines praktisch nutzbaren Quantencomputers weiterhin als sicher gelten.

3.1.2 Asymmetrische Kryptographie

In der asymmetrischen Kryptographie werden zur Ver- und Entschlüsselung unterschiedliche Schlüssel verwendet. Die asymmetrische Kryptographie ist erst seit der Entdeckung des Merkle-Puzzle bekannt und wird seitdem erforscht (siehe auch Kapitel 4). Asymmetrische Kryptographie basiert auf Public-Key-Verfahren. Jede an der Kommunikation beteiligte Partei besitzt ein eigenes Schlüsselpaar, bestehend aus einem öffentlichen und einem privaten Schlüssel. Das Schlüsselpaar ist dabei zusammengehörig und wird beispielsweise aus einer Zufallszahl erzeugt. Der private Schlüsselteil muss dabei geheimgehalten werden, während der andere Teil veröffentlicht werden muss. Der öffentliche Schlüssel darf dabei keine Informationen über den privaten Schlüsselteil preisgeben, da die Berechnung des privaten Schlüssel aus dem öffentlichen Schlüsselteil die gesamte Kommunikation kompromittieren würde.

Wenn zwei Parteien A und B nun verschlüsselt kommunizieren möchten, verwendet B den öffentlichen Schlüsselteil von A zur Verschlüsselung einer Nachricht. Aufgrund der mathematischen Zusammengehörigkeit des Schlüsselpaares kann A den privaten Teil ihres Schlüssels zur Entschlüsselung der Nachricht verwenden, während Unbeteiligte den Inhalt der Information nicht erfahren können. Die Schwierigkeit der Verfahren basiert unter anderem auch darauf, dass Partei B trotz Kenntnis des Klartextes, des öffentlichen Schlüssels von A und dem aus beidem erzeugten Geheimtextes keine Rückschlüsse auf den privaten Schlüssel von A ziehen kann.

Ein bekanntes Verfahren eines asymmetrischen Kryptosystems ist der Schlüsselaustausch nach DH, der in Kapitel 4.1 genauer beschrieben wird. Mit dem Aufkommen von kryptographischen Berechnungen auf elliptischen Kurven - der Elliptic Curve

Cryptography (ECC) - wurde auch das DH-Verfahren auf elliptische Kurven übertragen und wird in diesem Zusammenhang Elliptic Curve Diffie Hellman (ECDH) genannt. Dieses wird in Kapitel 4.2 beschrieben.

Asymmetrische Schlüsselaustauschverfahren basieren in der Regel auf dem als mathematisch schweres Problem angesehenen DLP bzw. dem damit verwandten Elliptic Curve Discrete Logarithm Problem (ECDLP). Mit dem Aufkommen praktisch nutzbarer Quantencomputer geht man davon aus, dass dieses Problem in polynomieller Zeit lösbar ist. Für die asymmetrischen Verfahren bedeutet dies, dass diese auch mit höheren Schlüssellängen nicht mehr ausreichend Sicherheit bieten und als unsicher gelten. Das Gebiet der PQC beschäftigt sich daher hauptsächlich mit der Entwicklung von asymmetrischen Verfahren, die auch gegen einen Angreifer mit einem Quantencomputer ausreichend Sicherheit gewährleisten können. Das US-amerikanische NIST hat im Jahr 2016 eine Ausschreibung zur PQC gestartet, um solche Verfahren zu untersuchen und zu standardisieren.

3.2 Post-Quanten-Kryptographie

In der PQC werden Verfahren untersucht, die auch nach der Entwicklung eines praktisch einsetzbaren Quantencomputers eine vertrauliche Kommunikation gewährleisten können. Im Rahmen der NIST-Ausschreibung wurden dabei verschiedene Verfahren entwickelt und vorgeschlagen, die sich aufgrund ihrer mathematischen Grundlagen in fünf Kategorien einteilen können [4]. Dabei bieten die Kategorien unterschiedliche Vor- und Nachteile und könnten zukünftig in verschiedenen Anwendungsgebieten eine Rolle spielen [5].

3.2.1 Multivariate Verfahren

Multivariate Verfahren basieren auf nichtlineare Gleichungssystemen mit mehreren Variablen über endlichen Körpern. Eine Gleichung, die als öffentlicher Schlüssel zur Verschlüsselung einer Nachricht verwendet wird, kann nur durch Kenntnis des privaten Schlüssels gelöst werden.

Das bekannteste multivariate Verfahren ist das „Unbalanced Oil and Vinegar“-Verfahren.

3.2.2 Codebasierte Verfahren

Codebasierte Verfahren basieren auf einem Fehlerkorrekturalgorithmus, der zur Verschlüsselung einer Nachricht absichtlich fehlerhafte Code erzeugt, die nur mit Kenntnis des privaten Schlüssels korrigiert werden können. Eine Entschlüsselung ohne Kenntnis der Fehlerkorrektur ist nicht möglich.

Beim McEliece-Verfahren handelt es sich um ein auf dem Goppa-Code basierendes Verfahren zur Fehlerkorrektur.

3.2.3 Hashbasierte Verfahren

Bei den hashbasierten Post-Quanten-Verfahren bilden klassische Hashverfahren wie der Secure Hash Algorithm (SHA) die Grundlage. Hierbei gründet sich die Sicherheit der Verfahren auf der Schwierigkeit, Kollisionen von Hashfunktionen zu finden. Eine Nachricht wird mithilfe einer Einwegfunktion in einen eindeutigen Hash umgewandelt. Zu jeder Nachricht gibt es genau einen Hashwert, zu einem Hashwert gehören theoretisch jedoch umgekehrt mehrere Nachrichten. Dennoch ist es mathematisch schwer, bei einer gegebenen Nachricht mit zugehörigem Hashwert eine zweite Nachricht zu finden, die auf den gleichen Hashwert abbildet.

In der Regel werden hashbasierte Verfahren für die Erzeugung von Signaturen verwendet, um die Authentizität einer Nachricht feststellen zu können. Ein hashbasiertes Verfahren in der Post-Quanten-Kryptographie ist das SPHINCS-Schema.

3.2.4 Gitterbasierte Verfahren

Die Grundlage von gitterbasierten Verfahren bilden unabhängige lineare Vektoren in einem n-dimensionalen Raum. In einem Vektorraum existieren mehrere mathematische Probleme, die für gitterbasierte Verfahren verwendet werden, wie das Closest Vector Problem und das Shortest Vector Problem.

Das in der NIST-Ausschreibung eingereicht NTRUEncrypt-Verfahren bildet ein gitterbasiertes Verfahren.

3.2.5 Isogeniebasierte Verfahren

Isogeniebasierte Verfahren sind die jüngsten der Post-Quanten-Verfahren und daher noch sehr wenig erforscht. Sie basieren auf Abbildungen zwischen elliptischen Kurven in endlichen Räumen. Häufig werden zur Veranschaulichung isogeniebasierter Verfahren Isogeniegraphen skizziert, wobei die Knoten der Graphen einzelne elliptische Kurven darstellen, während die Kanten die Isogenien veranschaulichen. Der Algorithmus basiert dann auf dem Finden eines Pfades zwischen den einzelnen Punkten.

Ein vielversprechender Kandidat, um ein klassisches DH-Verfahren zu ersetzen, stellt dabei der SIDH dar. Die isogeniebasierten Schlüsselaustauschverfahren werden in Kapitel 4.3 ausführlicher beschrieben und sind die Grundlage dieser Arbeit.

3.2.6 Vergleich

Die verschiedenen Verfahren der PQC sind aufgrund ihrer mathematischen Grundlagen für verschiedene Anwendungsgebiete mehr oder weniger gut geeignet. In ihrer Masterthesis hat Jennifer Ritz die Kategorien der Post-Quantum-Verfahren miteinander verglichen und in ihrer Anwendbarkeit gegenübergestellt. Dabei wurde unter der Sicherheit der Verfahren ebenfalls berücksichtigt, wie lange die Verfahren bereits bekannt sind und kryptographisch untersucht werden. [6, s. Kapitel 3.3.6]

Verfahren	**Sicherheit**	**Performanz**	**Schlüssellänge**	**Anwendungsgebiet**
Multivariat	0	0	-	Signaturerstellung
Code	+	+	-	Verschlüsselung
Hash	+	0	0	Signaturerstellung
Gitter	0	+	+	Verschlüsselung, Signaturerstellung
Isogenie	-	-	+	Schlüsselaustausch

Abb. 3.1: Vergleich der Post-Quantum-Verfahren [6, s. Tabelle 4]

4 Schlüsselaustauschverfahren

Bis in die 1970er Jahre existierten ausschließlich Verschlüsselungsverfahren, bei denen der Schlüssel für die Ver- und Entschlüsselung identisch ist. Für eine geheimzuhaltende Kommunikation zwischen zwei Parteien bedeutete dies, dass der Schlüssel zuvor beiden Parteien bekannt gemacht werden musste. Für zwei Parteien, die sich vor der Kommunikation nicht kannten und Daten ausschließlich über potenziell unsichere Kommunikationskanäle austauschen können, stellte dies eine unüberwindbare Hürde für vertraulichen Nachrichtenaustausch dar.

Der US-amerikanische Wissenschaftler Ralph C. Merkle entdeckte 1974 mit dem *Merkle Puzzle* die Grundlage für assymmetrische Verschlüsselungsverfahren, bei denen sich Sende- und Empfängerpartei vor dem Nachrichtenaustausch nicht kennen müssen und sich über ein unsicheres Kommunikationsmedium auf ein gemeinsames Geheimnis einigen können. Das Merkle Puzzle findet zwar keine praktische Anwendung, konnte aber zeigen, dass es eine Möglichkeit für zwei Parteien gibt, sich in kurzer Zeit auf ein gemeinsames Geheimnis zu einigen und eventuelle Mithörer ein Vielfaches dieser Zeit benötigen, um das Geheimnis zu errechnen. Das Merkle Puzzle gilt daher als der Beginn der Schlüsselaustauschverfahren [7].

4.1 Klassischer Diffie-Hellman

Kurze Zeit nach der Entdeckung des Merkle Puzzle veröffentlichten die US-amerikanischen Kryptologen Whitfield Diffie und Martin Hellman mit dem DH-Verfahren einen praktisch verwendbaren Algorithmus, nach welchem sich zwei Parteien über einen unsicheren Kanal auf ein gemeinsames Geheimnis einigen können. Hierfür werden zur Ver- und Entschlüsselung von Nachrichten zwei verschiedene Schlüssel verwendet, einem öffentlich bekannten Schlüssel und einem privaten Schlüssel. Beide Schlüssel zusammen ergeben das eigene *Schlüsselpaar* [8].

Beim klassischen DH einigen sich die Parteien A und B auf eine (große) Primzahl p und eine natürliche Zahl $g < p$ als Basis. Partei A wählt sich nun eine Zufallszahl $a < p$ als ihren privaten Schlüssel und berechnet ihren öffentlichen Schlüssel als

$A = g^a \mod p$. Partei B wählt sich die Zufallszahl $b < p$ und berechnet $B = g^b \mod p$. Beide Parteien tauschen nun ihren errechneten öffentlichen Schlüssel untereinander aus. Als gemeinsames Geheimnis berechnet A nun $K = B^a = g^{ba} = g^{ab} \mod p$ und Partei B $K = A^b = g^{ab} = g^{ba} \mod p$. Die Vorgehensweise ist in 4.1 veranschaulicht.

Öffentlich bekannte Parameter	Primzahl p, natürliche Zahl $g < p$	
	Alice	**Bob**
zufälliges Geheimnis wählen	$a \in \{1, .., p-1\}$	$b \in \{1, .., p-1\}$
öffentliche Schlüssel berechnen	$A = g^a \mod p$	$B = g^b \mod p$
Datenaustausch	$A \longrightarrow$	$\longleftarrow B$
privaten Schlüssel berechnen	$K = B^a$	$K = A^b$

Abb. 4.1: Diffie-Hellman Schlüsselaustausch

Beispiel: Alice und Bob einigen sich darauf, die Primzahl $p = 23$ und die Basis $g = 5$ für ihren Schlüsselaustausch zu verwenden. Alice wählt sich die Zufallszahl $a = 7$ und berechnet $A = g^a \mod p = 5^7 \mod 23 = 17$. Bob wählt sich die Zufallszahl $b = 11$ und berechnet $B = g^b \mod p = 5^{11} \mod 23 = 22$. Beide übermitteln ihren öffentlichen Teil des Schlüssels A bzw. B. Anschließend berechnet Alice das gemeinsame Geheimnis als $K = B^a \mod p = 22^7 \mod 23 = 22$, während Bob das gemeinsame Geheimnis berechnet als $K = A^b \mod p = 17^{11} \mod 23 = 22$. Das gemeinsame Geheimnis $K = 22$ kann nun zur Nachrichtenverschlüsselung über einen unsicheren Kommunikationskanal verwendet werden.

Die Sicherheit des klassischen DH basiert darauf, dass bei gegebenem g, a leicht ist zu bestimmen $M = g^a$. Umgekehrt ist es mathematisch schwer, bei gegebenem M, g den Exponenten a zu bestimmen. Dieses mathematische Problem wird *Computational Diffie-Hellman* genannt und ist eng verwandt mit dem DLP.

4.2 Diffie-Hellmann mit Elliptischen Kurven

Es hat sich gezeigt, dass DH nicht nur auf endlichen Körpern anwendbar ist, sondern sich verallgemeinert auf endlichen Gruppen anwenden lässt. So schlug Neil Koblitz 1985 vor, Gruppen von Punkten auf elliptischen Kurven über endlichen Körpern für Schlüsselaustauschverfahren nach DH zu verwenden [9].

Aufgrund der in Kapitel 2.1 gezeigten Gruppenoperationen auf elliptischen Kurven lassen sich die DH-Operationen über endlichen Körpern einfach erweitern zu einem

ECDH. Statt sich auf einen endlichen Körper festzulegen, einigen sich die beteiligten Parteien auf eine Gruppe von Punkten einer elliptischen Kurve über einem endlichen Körper. So wird statt einer Primzahl p bzw. dem endlichen Körper $\mathbb{F}_p$ nun eine elliptische Kurve E über dem endlichen Körper $\mathbb{F}_p$ Teil der öffentlich bekannten Parameter. Statt einer Ganzzahl $g \in \mathbb{F}_p$ einigt man sich außerdem auf einen Punkt $G \in E$. Idealerweise handelt es sich bei G um einen Generatorpunkt von E, also einen Punkt, aus dem sich durch Skalarmultiplikation alle anderen Punkte von E über F_p erzeugen lassen.

Öffentlich bekannte Parameter	Elliptische Kurve E, Generatorpunkt $G \in E$, Modul n	
	Alice	**Bob**
zufälliges Geheimnis wählen	$a \in \{1, .., n-1\}$	$b \in \{1, .., n-1\}$
öffentliche Schlüssel berechnen	$A = a \cdot G \mod n$	$B = b \cdot G \mod n$
Datenaustausch	$A \longrightarrow$	$\longleftarrow B$
privaten Schlüssel berechnen	$K = a \cdot B$	$K = b \cdot A$

Abb. 4.2: Elliptic Curve Diffie-Hellman

Vergleicht man das Schema 4.2 mit dem ursprünglichen DH, fällt schnell auf, dass die Gruppenoperationen ähnlich sind. Tatsächlich liegt für die Sicherheit von ECDH ein Problem zugrunde, das eng verwandt mit dem diskreten Logarithmus ist und ECDLP genannt wird. Dennoch zeigt sich, dass Operationen im Rahmen von ECDH schneller ausführen lassen, da auf elliptischen Kurven anstelle von Potenzen lediglich Punktmultiplikationen durchgeführt werden müssen. Über elliptischen Kurven lassen sich daher Kryptosysteme entwickeln, die in gleichbleibender Zeit mit größeren Schlüssellängen rechnen können.

Beispiel: Alice und Bob einigen sich darauf, die Kurve $E: \ y^2 = x^3 + 5x^2 + x$ über dem endlichen Körper $\mathbb{F}_p$ mit $p = 23$ für den Schlüsselaustausch zu verwenden. Als Generatorpunkt auf der Montgomery-Kurve verwenden beide den Punkt $G = (5 : 5)$. Alice wählt sich die Zufallszahl $a = 7$ und berechnet $A = a \cdot G \mod p = 7 \cdot (5 : 5) \mod 23 = (17 : 21)$. Bob wählt sich die Zufallszahl $b = 11$ und berechnet $B = b \cdot G \mod p = 11 \cdot (5 : 5) \mod 23 = (14 : 9)$. Beide übermitteln ihren öffentlichen Teil des Schlüssels A bzw. B. Anschließend berechnet Alice das gemeinsame Geheimnis als $K = a \cdot B \mod p = 7 \cdot (14 : 9) \mod 23 = (19 : 9)$, während Bob das gemeinsame Geheimnis berechnet als $K = b \cdot A \mod p = 11 \cdot (17 : 21) \mod 23 = (19 : 9)$. Das gemeinsame Geheimnis $K = (19 : 9)$ kann nun zur Nachrichtenverschlüsselung über einen unsicheren Kommunikationskanal verwendet werden.

4.3 Isogeniebasierter Schlüsselaustausch

Bei einem isogeniebasierten Schlüsselaustausch einigen sich die Parteien Alice und Bob auf eine öffentlich bekannte Primzahl p und eine Ausgangskurve E_0 über dem quadratischen Erweiterungskörper $\mathbb{F}_{p^2}$. Ebenfalls öffentlich bekannt sind die Generatorpunkte P_a, Q_a, P_b und Q_b mit $P_a, Q_a, P_b, Q_b \in E_0$. Dabei müssen folgende Bedingungen erfüllt sein.

- Primzahl $p = l_A^{e_A} \cdot l_B^{e_B} \pm 1$
- $E_0 : \ y^2 = x^3 + Ax^2 + x$
- P_a, Q_a sind Punkte auf E_0 der Ordnung $l_A^{e_A}$
- P_b, Q_b sind Punkte auf E_0 der Ordnung $l_B^{e_B}$

Häufig werden für die Parameter l_A und l_B die Werte $l_A = 2$ und $l_B = 3$ verwendet, da diese nach aktueller Sicht mit entsprechenden Potenzen e_A und e_B ausreichend sind. Die Berechnung der Isogenien wird außerdem vereinfacht, da sich die aus den Punkten erzeugten Isogenien so stets auf die Basis 2 bzw. 3 beziehen. In der folgenden Beschreibung sollen daher diese Werte so verwendet werden.

Der quadratische Erweiterungskörper $\mathbb{F}_{p^2}$ wird deshalb gewählt, weil somit sichergestellt wird, dass der endliche Körper die Charakteristik p besitzt. Dies bedeutet, dass man das multiplikativ neutrale Element p-mal multiplizieren muss, um das additiv neutrale Element zu erhalten. Durch die Wahl des quadratischen Erweiterungskörpers erhalten Punkte und Parameter ein komplexes Element i, sodass beispielsweise Punktkoordinaten stets in der Form $(X_1 i + X_2 : Y_1 i + Y_2)$ angegeben werden.

Alice wählt sich nun eine geheime Ganzzahl $k_a \in \{1, .., (2^a - 1)\}$ und berechnet ihren geheimen Startpunkt $R_a = P_a + k_a Q_a$. Der Punkt R_a hat die Ordnung 2^{e_A} auf E_0. Durch mehrfache Punktverdopplung erzeugt sich Alice dann einen Generatorpunkt G_a der Ordnung 2 auf E_0. Durch e_A-fache Komposition der mithilfe von Velu's Formeln berechneten 2-Isogenien erzeugt Alice ihre Abbildung $\phi_a : \ E_0 \mapsto E_a$. Sie überträgt außerdem die Punkte P_b und Q_b mithilfe von Velu's Formeln und erzeugt so ihren öffentlichen Schlüssel $K_{pubA} = (\phi_a(E_0), \phi_a(P_b), \phi_a(Q_b))$.

Gleichzeitig wählt Bob eine geheime Ganzzahl $k_b \in \{1, .., (3^b - 1)\}$ und berechnet seinen geheimen Startpunkt $R_b = P_b + k_b Q_b$. Der Punkt R_b hat die Ordnung 3^{e_B} auf E_0. Durch mehrfache Punktverdreifachung erzeugt sich Bob dann einen Generatorpunkt G_b der Ordnung 3 auf E_0. Durch e_B-fache Komposition der mithilfe von Velu's

Formeln berechneten 3-Isogenien erzeugt Bob seine Abbildung $\phi_b : \ E_0 \mapsto E_b$. Er überträgt außerdem die Punkte P_a und Q_a mithilfe von Velu's Formeln und erzeugt so seinen öffentlichen Schlüssel $K_{pubB} = (\phi_b(E_0), \phi_b(P_a), \phi_b(Q_a))$.

Zur Berechnung des gemeinsamen Geheimnisses startet Alice nun auf der elliptischen Kurve $\phi_b(E_0)$ aus Bob's öffentlichem Schlüssel und erzeugt sich wiederum ihren Punkt $R_a = \phi_b(P_a) + k_a\phi_b(Q_a)$ der Ordnung 2^{e_A} auf der Kurve $\phi_b(E_0)$. Sie berechnet nun wie bei der Erzeugung ihres öffentlichen Schlüssels ihre Abbildung ϕ_{ab} als e_A-fache Komposition der 2-Isogenien auf $\phi_b(E_0)$.

Bob berechnet das gemeinsame Geheimnis, indem er ϕ_{ba} als e_B-fache Komposition der 3-Isogenien auf $\phi_a(E_0)$ erzeugt. Hierbei geht er ähnlich wie bei der Erzeugung seines öffentlichen Schlüssels vor und erzeugt den Punkt $R_b = \phi_a(P_b) + k_b\phi_a(Q_b)$ der Ordnung 3^{e_B} auf $\phi_a(E_0)$.

Die Ergebniskurven ϕ_{ab} und ϕ_{ba} sind nicht unbedingt *gleich*, es lässt sich aber zeigen, dass beide Kurven zueinander *isomorph* sind. Zwei zueinander isomorphe elliptische Kurven haben die gleiche j-Invariante. Dieser Wert wird beim supersingulären Schlüsselaustauschverfahren als gemeinsames Geheimnis verwendet.

$K_{ab} = j(\phi_{ab}) = j(\phi_{ba})$

Öffentlich bekannte Parameter	Primzahlen l_A, l_B und $p = l_A^{e_A} \cdot l_B^{e_B} \pm 1$, Eine supersinguläre elliptische Kurve E über dem endlichen Körper $\mathbb{F}_{p^2}$, Eine Basis $\langle P_a, Q_a \rangle$ auf der Kurve E mit der Ordnung $l_A^{e_A}$, Eine Basis $\langle P_b, Q_b \rangle$ auf der Kurve E mit der Ordnung $l_B^{e_B}$	
	Alice	**Bob**
zufälliges Geheimnis wählen	$k_a \in \{1, .., l_A^{e_A}\}$	$k_b \in \{1, .., l_B^{e_B}\}$
geheimen Generatorpunkt berechnen	$R_a = P_a + a \cdot Q_a$	$R_b = P_b + b \cdot Q_b$
geheimen Isogenie berechnen	$\alpha : E \mapsto E_A = E/\langle R_a \rangle$	$\beta : E \mapsto E_B = E/\langle R_b \rangle$
Datenaustausch	$E_A, \alpha(P_b), \alpha(Q_b)$	$E_B, \alpha(P_a), \alpha(Q_a)$
gemeinsames Geheimnis berechnen	$E/\langle R_a, R_b \rangle = E_B \mapsto E_{AB}$	$E/\langle R_a, R_b \rangle = E_A \mapsto E_{AB}$

Abb. 4.3: Protokoll eines Supersingular Isogeny Diffie-Hellmann

Beispiel: Alice und Bob einigen sich auf eine Primzahl $p = l_A^{e_A} \cdot l_B^{e_B} - 1$ mit $l_A = 2$, $e_A = 8$, $l_B = 3$ und $e_B = 5$ ($p = 2^8 \cdot 3^5 - 1 = 62207$), sowie die supersinguläre elliptische Kurve $E_0 : \ y^2 = x^3 + Ax^2 + x$ mit $A = 0$ ($E_0 : y^2 = x^3 + x$) über dem

quadratischen Erweiterungskörper $\mathbb{F}_{p^2}$. Öffentlich bekannt sind außerdem die Punkte P_a, Q_a der Ordnung $l_A^{e_A} = 2^8 = 256$ auf E_0 und P_b, Q_b der Ordnung $l_B^{e_B} = 3^5 = 243$ auf E_0:

$P_a = (29060i + 10030 : 53689i + 58342)$
$Q_a = (19671i + 11802 : 6880i + 20353)$
$P_b = (44141i + 60177 : 45292i + 15694)$
$Q_b = (54009i + 60928 : 2011i + 6309)$

Alice wählt die Ganzzahl $k_a = 2 \in \{1, .., (2^a - 1)\}$, Bob wählt die Ganzzahl $k_b = 3 \in \{1, .., (3^b - 1)\}$. Damit erzeugt Alice den Punkt $R_a = P_a + k_a Q_a$ als Punkt der Ordnung 256, Bob erzeugt den Punkt $R_b = P_b + k_b Q_b$ als Punkt der Ordnung 243 auf E_0.

$R_a = P_a + k_a Q_a = (29060i + 10030 : 53689i + 58342)$
$R_b = P_b + k_b Q_b = (14652i + 17701 : 25882i + 34638)$

Um einen Generatorpunkt G_a der Ordnung 2 zur Berechnung der 2-Isogenie nach Velu's Formel zu erhalten, muss Alice den Punkt R_a mit 128 multiplizieren: $G_a = 128 \cdot R_a$ und erzeugt aus G_a ihre erste 2-Isogenie ϕ_{a_1}. Nach der Übertragung der Punkte $R'_a = \phi_{a_1}(R_a)$, $P'_b = \phi_{a_1}(P_b)$ und $Q'_b = \phi_{a_1}(Q_b)$ hat der Punkt R'_a die (halbierte) Ordnung 128 auf der durch die Isogenie ϕ_{a_1} abgebildeten Kurve, während P'_b und Q'_b die Ordnung 243 behalten (da 128 kein Teiler der Ordnung ist). Diese Schritte wiederholt Alice so lange mit den Punkten R'_a, P'_b und Q'_b, bis R'_a die Ordnung 2 hat und somit in einem letzten Schritt auf das Identitätselement $\phi_{a_m}(R'_a) = \mathcal{O}$ abbildet. Am Ende erhält Alice das Ergebnis der 2^8-Isogenie als Komposition der acht erzeugten 2-Isogenien $\phi_a = (\phi_{a_m} \circ \phi_{a_{m-1}} \circ ... \circ \phi_{a_1})$. Als Ergebniskurve erhält Alice die Abbildung der Isogenie $E_A = \phi_a(E_0)$. Ihr öffentlicher Schlüssel ergibt sich dann als $K_{pubA} = (E_A, \phi_a(P_b), \phi_a(Q_b))$.

$E_A : y^2 = x^3 + (41880i + 14601)x + (617i + 12939)$
$j(E_A) = 48872i + 16634$
$\phi_a(P_b) = (5089i + 46144 : 58522i + 8338)$
$\phi_a(Q_b) = (59489i + 35968 : 24404i + 1190)$

Bob erhält seinen Generatorpunkt G_b der Ordnung 3 zur Berechnung der 3-Isogenie nach Velu's Formel, indem er den Punkt R_b mit 81 multipliziert: $G_b = 81 \cdot R_b$. Daraus erzeugt Bob seine erste 3-Isogenie ϕ_{b_1} und überträgt die Punkte $R'_b = \phi_{b_1}(R_b)$, $P'_a = \phi_{b1}(P_a)$ und $Q'_a = \phi_{b_1}(Q_a)$. Der Punkt P'_b erhält so die Ordnung 81 auf ϕ_{b_1}, während P'_a und Q'_a die Ordnung 256 behalten (da 81 kein Teiler von 256 ist). Diese Schritte wiederholt Bob so lange mit den Punkten R'_b, P'_a und Q'_a, bis R'_b die Ordnung

3 hat und somit in einem letzten Schritt auf das Identitätselement $\phi_{b_n}(R'_b) = \mathcal{O}$ abbildet. Am Ende erhält Bob das Ergebnis der 3^5-Isogenie als Komposition der fünf erzeugten 3-Isogenien $\phi_b = (\phi_{b_n} \circ \phi_{b_{n-1}} \circ ... \circ \phi_{b_1})$. Als Ergebniskurve erhält Bob die Abbildung der Isogenie $E_B = \phi_b(E_0)$. Sein öffentlicher Schlüssel ergibt sich als $K_{pubB} = (E_B, \phi_b(P_a), \phi_b(Q_a))$.

$E_B : y^2 = x^3 + (54213i + 47142)x + (45584i + 45420)$
$j(E_B) = 59055i + 47550$
$\phi_b(P_a) = (12710i + 47747 : 33758i + 5173)$
$\phi_b(Q_a) = (38353i + 34841 : 14473i + 47475)$

Zur Erzeugung des gemeinsamen Geheimnisses nimmt Alice als Startkurve die Ergebniskurve aus Bob's öffentlichem Schlüssel ϕ_b sowie die von Bob berechneten Punkte $\phi_b(P_a)$ und $\phi_b(Q_a)$ zur Erzeugung ihres Startpunktes $R_a = \phi_b(P_a) + k_a\phi_b(Q_a)$ der Ordnung 256 auf E_B. Anschließend erzeugt sie ihre 2^8-Isogenie als Komposition der acht 2-Isogenien auf E_B und erhält E_{AB} als Abbildung von ϕ_{ab}.

$E_{AB} : y^2 = x^3 + (35789i + 56578)x + (61730i + 8760)$
$j(E_{AB}) = 22319i + 23990$

Bob nimmt als Startkurve die Ergebniskurve aus Alice's öffentlichem Schlüssel ϕ_a sowie die von Alice berechneten Punkte $\phi_a(P_b)$ und $\phi_a(Q_b)$ zur Erzeugung seines Startpunktes $R_b = \phi_a(P_b) + k_b\phi_a(Q_b)$ der Ordnung 243 auf E_A. Er erzeugt seine 3^5-Isogenie als Komposition der fünf 3-Isogenien auf E_A und erhält E_{BA} als Abbildung von ϕ_{ba}.

$E_{BA} : y^2 = x^3 + (35789i + 56578)x + (61730i + 8760)$
$j(E_{BA}) = 22319i + 23990 = j(E_{AB})$

Es sei angemerkt, dass im vorliegenden Beispiel für die Ergebniskurven ebenfalls gilt $E_{AB} = E_{BA}$. Dies ist jedoch häufig nicht der Fall, sodass nach dem Protokoll stets die j-Invarianten verglichen werden.

5 Tests der Schlüsselaustauschverfahren

5.1 Allgemeines Setup

Die Tests werden auf einer virtuellen Maschine durchgeführt. Das zugrundeliegende Betriebssystem ist ein Ubuntu 20.04.2 LTS (focal). Als Prozessor wird ein Intel(R) Core(TM) i5 Dual Core mit einer Taktrate von 1,2 GHz und einer Cachegröße von 4096 KB verwendet. Als Arbeitsspeicher stehen 4 GB virtualisierter RAM zur Verfügung.

Für die Schlüsselaustauschverfahren nach DH und ECDH wurde die Crypto++-Library in der Version 8.5 verwendet [10]. Für SIDH wurde die von Microsoft's PQCrypto-Team veröffentlichte SIDH-Bibliothek betrachtet [11]. Für Commutative Supersingular Isogeny Diffie Hellman (CSIDH) wurde die vom CSIDH-Team veröffentlichte Referenzimplementierung betrachtet [12].

Die Crypto++-Library wurde heruntergeladen und mittels des Befehls `make install` installiert. Die bereitgestellten Programmbibliotheken können somit mittels `#include` in eigenem Quelltext verwendet werden.

Für die Zeitmessung der Verfahren werden unterschiedliche Feldgrößen und damit unterschiedlich große Schlüssellängen verwendet, um aufzuzeigen, welche Verfahren anhand der Rechenoperationen für ein gleichbleibendes Sicherheitsniveau ein effizienteres Zeitverhalten aufweisen. In der Regel wird die Sicherheit von Verschlüsselungsverfahren mit dem symmetrischen Verschlüsselungsverfahren AES verglichen. Wenn also ein Verfahren ein *klassisches Sicherheitsniveau* von 128 Bit erreicht, ist dieses vergleichbar mit dem AES-Verfahren mit einer Schlüssellänge von 128 Bit. Die Vergleichswerte des NIST sind in [13, vgl. Table 2] nachzulesen. Eine gute Übersicht über internationale Empfehlungen von Schlüssellängen im Vergleich bietet [14].

Um in den Tests ein vergleichbares klassisches Sicherheitslevel von 128 Bit zu erreichen, werden die folgenden Parameter für die Verfahren verwendet: Das Diffie-Hellman-Verfahren wird berechnet über einen endlichen Körper über einer Primzahl der Länge von 3072 Bit [vgl. auch 15], der endliche Körper für ECC ist definiert

über eine Primzahl der Größe von 256 Bit [16, s. Kapitel 4.1], die Referenzimplementierung zu SIDHp503 [17, vgl. S.55] und ein endlicher Körper von 512 Bit für CSIDH [18].

Nicht in der Zeitanalyse enthalten sind die Zeiten, um erzeugte öffentliche Schlüssel zwischen den Parteien auszutauschen. Einerseits sind diese Zeiten abhängig von der Größe der Schlüssel, da bei größeren Schlüssellängen mehr Daten übertragen werden müssen, andererseits aber zu einem weitaus größeren Faktor vom gewählten Übertragungsmedium. Die Zeiten zum eigentlichen Austausch der Schlüssel sollen daher hier vernachlässigt werden.

5.2 Diffie-Hellman

Für die Untersuchung des Zeitverhaltens eines Schlüsselaustausches nach Diffie-Hellman wurde die Klasse DH der C++-Programmbibliothek Crypto++ verwendet (Dokumentation unter [19]).

Mithilfe der Crypto++-Bibliothek ist es ebenfalls möglich, eine zufällige Primzahl mit angegebener Bitgröße zu erzeugen. Dies ist für größere Bitlängen jedoch sehr unpraktikabel. So wird der Zeitaufwand zum Generieren einer 3072-Bit Primzahl auf einem Core 2 Duo mit etwa 18 Minuten angegeben [20, s. Überschrift 3 „Generating Parameters (MODP)“]. Um die Ausführungszeit für den Testlauf zu verringern, wurde daher auf die empfohlenen Parameter des RFC3526 der Internet Engineering Task Force (IETF) für einen 3072-Bit Diffie-Hellman-Schlüsselaustausch verwendet [21, s. Kapitel 4].

Die aus dem RFC3526 verwendete 3072-Bit Primzahl und die zugehörige Basis erzeugen in der Crypto++-Bibliothek private Schlüssel von 272 Bit und öffentliche Schlüssel von 3072 Bit, um einen geteilten Schlüssel von 3072 Bit zu erzeugen. Die Schlüsselerzeugung wurde 1000-mal ausgeführt, um einen aussagekräftigen Mittelwert bilden zu können.

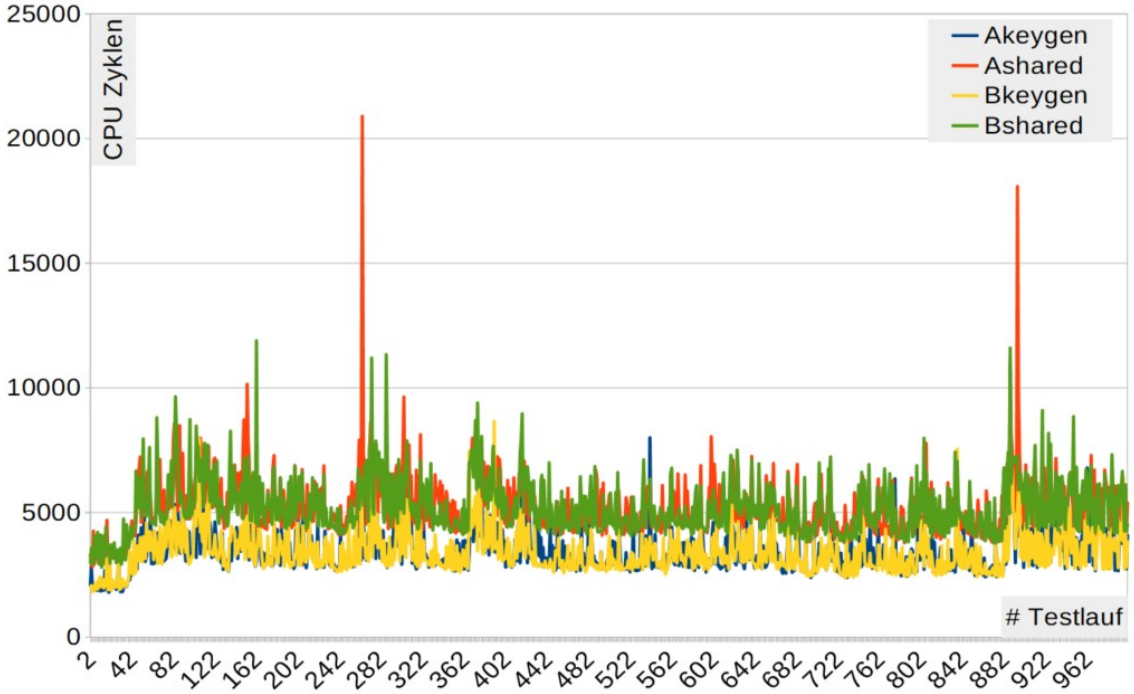

Abb. 5.1: Datenreihe des DH-Schlüsselaustauschs

In Abb. 5.1 wird über 1000 Testläufe angegeben, wie viele CPU-Zyklen die einzelnen Schritte des Schlüsselaustausches benötigen. Hierbei bezeichnen Akeygen und Bkeygen die CPU-Zeit, die Alice (A) und Bob (B) jeweils benötigen, um ihren privaten und öffentlichen Schlüssel zu berechnen. Ashared und Bshared sagen aus, wie lange Alice und Bob jeweils für die Berechnung des gemeinsamen Geheimnisses benötigen.

5.3 Elliptic Curve Diffie-Hellman

Für die Untersuchung des Zeitverhaltens eines Schlüsselaustausches nach Elliptic Curve Diffie-Hellman wurde die standardisierte Curve25519 aus der Programmbibliothek Crypto++ verwendet (Dokumentation unter [22]).

Die Curve25519 wurde 2005 von Daniel Bernstein veröffentlicht [23]. Die Curve25519 erzeugt Schlüssel mit einer Länge 256 Bit und bietet somit ein klassisches Sicherheitsniveau von 128 Bit. Die Parameter sind nicht patentgeschützt und können daher frei verwendet werden. So basieren die Schlüsselaustauschverfahren der Messenger-App Signal und eine Implementierung der Secure Shell (SSH) auf dem Curve25519-Kryptosystem. Namensgebend für das Kryptosystem ist der endliche Körper $\mathbb{F}_p$ mit

$p = 2^{255} - 19$, über welchem die elliptische Kurve $E: \ y^2 = x^3 + 486662x^2 + x$ definiert ist. Die IETF hat im RFC7748 außerdem den Generatorpunkt $G = (X : Y)$ mit X und Y als bekannten Parametern veröffentlicht [16].

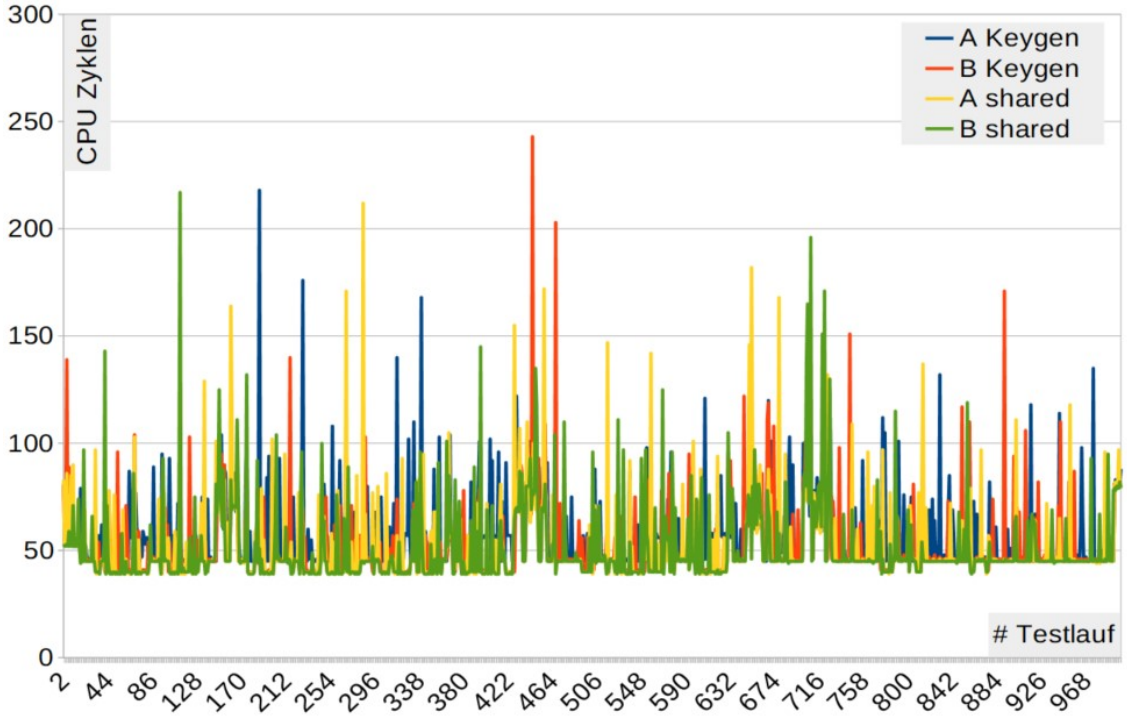

Abb. 5.2: Datenreihe des ECDH-Schlüsselaustauschs mit der Curve25519

5.4 Supersingular Isogeny Diffie-Hellmann (SIDH)

Im Rahmen der fortschreitenden Forschung an Quantencomputern hat das NIST im Rahmen ihrer Standardisierungsbestrebungen dazu aufgerufen, quantum-resistente Schlüsselaustauschverfahren einzureichen. Diese werden seit Dezember 2016 evaluiert. Zu den Kandidaten der Public-Key Schlüsselaustauschverfahren wurde auch die Referenzimplementierung SIKE eingereicht, die eine Umsetzung eines SIDH-Schlüsselaustauschverfahrens ist und ihrerseits das erste praktisch nutzbare isogeniebasierte Schlüsselaustauschverfahren darstellt. Im Folgenden wird die Referenzimplementierung von Microsoft Research für SIDHp503 betrachtet [11].

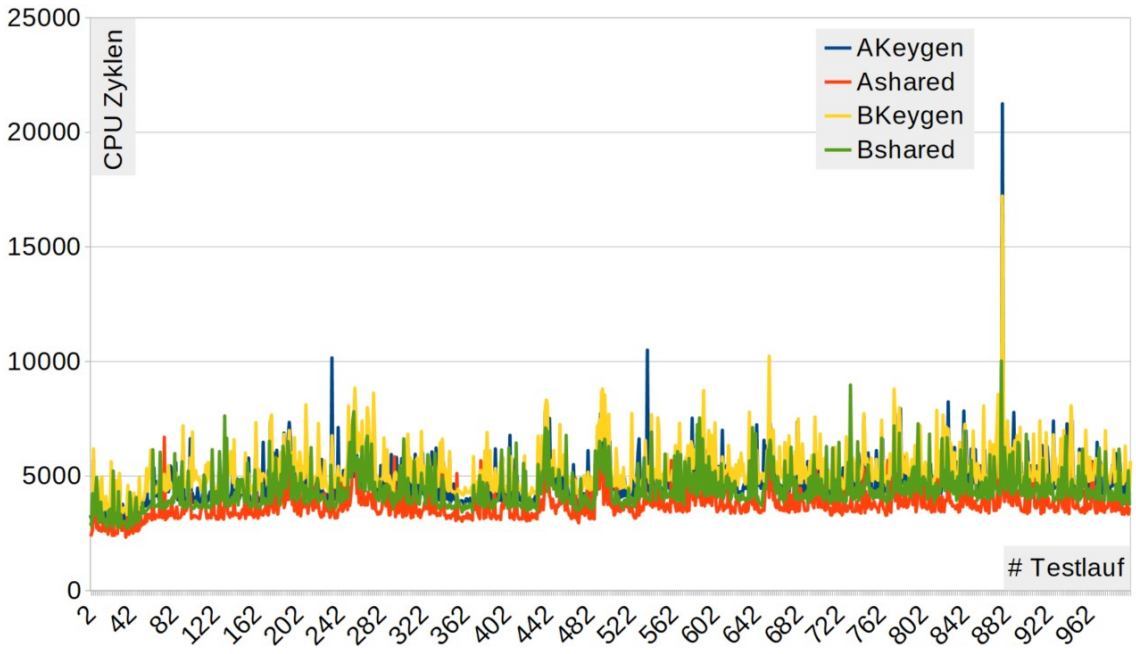

Abb. 5.3: Datenreihe des SIDH-Schlüsselaustauschs

5.5 Commutative Supersingular Isogeny Diffie-Hellmann (CSIDH)

Während SIKE aufgrund verschiedener Effizienzsteigerungen stetig schneller geworden ist, sehen die Forscherinnen und Forscher von CSIDH ein großes Problem in der Schlüssellänge von SIKE. Um mit der Implementierung SIDHp503 ein klassisches Sicherheitsniveau von 128 Bit zu erreichen, werden Schlüssellängen von 3024 Bit benötigt. Dies ist vergleichbar mit den Schlüssellängen bei einem klassischen Diffie-Hellman-Verfahren.

Des Weiteren bemerkt das CSIDH-Team, dass Schlüssel aufgrund verschiedener Angriffe auf SIDH nicht wiederverwendet werden können.

CSIDH stellt mit einer vergleichsweise kurzen Schlüssellänge von 512 Bit und einer integrierten Validierung der verwendeten Schlüssel die Möglichkeit bereit, veröffentlichte Schlüssel für weitere isogeniebasierte Schlüsselaustausche zu verwenden. Desweiteren kann CSIDH aufgrund der kleinen Schlüssellängen auch für Protokolle eingesetzt werden, die kürzere Schlüssellängen benötigen oder bei denen Übertragungsgeschwindigkeiten priorisiert werden [18].

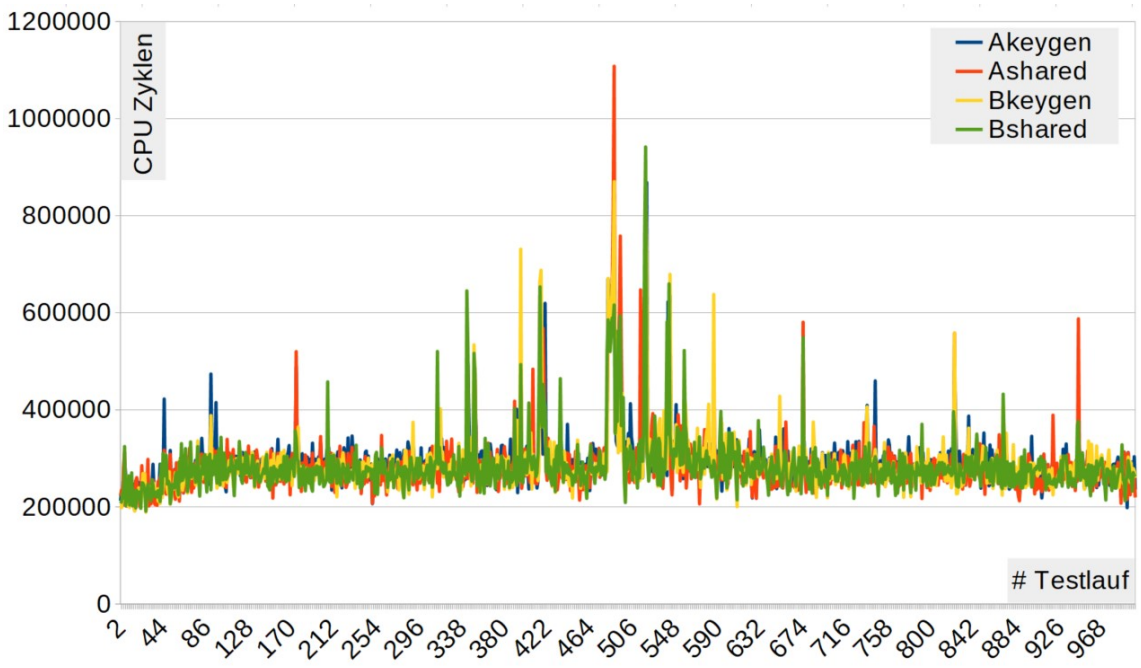

Abb. 5.4: Datenreihe des CSIDH-Schlüsselaustauschs

5.6 Vergleich

Alle Testläufe wurden 1000 Mal ausgeführt. In Abbildung 5.5 wurden aus den einzelnen Testergebnissen Mittelwerte gebildet, um diese miteinander ins Verhältnis setzen zu können. Die Gleitkommawerte wurden auf den nächsten Ganzzahlwert gerundet.

	Akeygen	**Ashared**	**Bkeygen**	**Bshared**
DH	3416	5224	3401	5213
ECDH	56	52	52	51
SIDH	4751	3830	5225	4395
CSIDH	293116	284781	285748	283809

Abb. 5.5: Mittelwerte der benötigten CPU-Zyklen während der Testläufe

Die Gegenüberstellung der Verfahren zeigt, dass der klassische Schlüsselaustausch nach Diffie-Hellman aufgrund der langsameren Rechenoperationen mit großen Schlüssellängen ein schlechtes Zeitverhalten besitzt. Demgegenüber sind die Berechnungen auf elliptischen Kurven weitaus zeiteffizienter und bieten auch mit kürzeren Schlüssellängen ein vergleichsweise hohes Sicherheitsniveau. Für höhere Sicherheitsniveaus stehen neben der Curve25519 weitere Implementierungen mit höheren Schlüssellängen zur Verfügung.

Das SIDH-Verfahren bietet eine ähnliche zeitliche Performance wie der klassische Diffie-Hellman, hat aber mit einer ähnlich großen Schlüssellänge ähnliche Nachteile. Außerdem bietet die SIDH-Implementierung ausschließlich flüchtige Schlüssel, sodass für jeden Schlüsselaustausch ein neuer Schlüssel generiert werden müssen. Das Verfahren CSIDH behebt beide Nachteile und bietet ein gleichbleibendes Sicherheitsniveau bei kürzeren Schlüsseln. CSIDH integriert außerdem die Validierung von öffentlichen Schlüsseln, sodass die Flüchtigkeit behoben ist. Diese Vorteile erkauft sich CSIDH mit einer weitaus geringeren Performance.

6 Risikoanalyse der Schlüsselaustauschverfahren

6.1 Allgemeines

Im vorigen Kapitel wurde das Zeitverhalten der verschiedenen Schlüsselaustauschverfahren bei gleichbleibendem Sicherheitsniveau analysiert und gegenübergestellt. Im folgenden Kapitel sollen die einzelnen Verfahren auf ihre Schwachstellen und ihre Angriffsvektoren untersucht werden. Im Vergleich sollen die einzelnen Angriffsvektoren auf die Verfahren gegenübergestellt und bewertet werden.

Einen Angriffsvektor stellt außerdem ein praktisch nutzbarer Quantencomputer dar. Während die klassischen Verfahren DH und ECDH auf dem Problem des diskreten Logarithmus basieren und damit von Quantencomputern in polynomieller Zeit gebrochen werden können, bieten isogeniebasierte Verfahren nach aktuellem Stand eine geringere Angriffsfläche für Quantencomputer. Dies wird in der Gegenüberstellung berücksichtigt.

6.2 Verfahren zur Risikoanalyse

In der IT-Sicherheitsbranche haben sich inzwischen unterschiedliche Verfahren etabliert, um Risiken der Systeme evaluieren und analysieren zu können. In der Risikoanalyse sollen so Bedrohungen abstrakt festgehalten werden, um geeignete Gegenmaßnahmen zu implementieren. Die Risikoanalyse soll außerdem dabei helfen, im Unternehmen eingesetzte Software und Tools ersetzen zu können, ohne das Sicherheitsniveau zu verringern.

Viele der in der IT-Sicherheit eingesetzten Risikoanalyseverfahren betrachten dabei gesamte IT-Systeme oder Prozesse in der IT. Als bekanntester Grundsatz der IT-Sicherheit bildet der sogenannte CIA-Dreiklang ("**C**onfidentiality" - Vertraulichkeit, "**I**ntegrity" - Integrität und "**A**vailability" - Verfügbarkeit) für die meisten Risikobewertungssysteme den ersten Ansatz. Jede Bedrohung ist somit mindestens einem dieser Punkte zugeordnet und kann anhand ihrer Schwere und Auswirkung bewertet werden.

Zur Bewertung von Bedrohungen existieren verschiedene Modelle, von denen drei hier vorgestellt werden.

6.2.1 DREAD

Das Bedrohungsmodell DREAD wurde nach den fünf Kategorien benannt, nach welchen Bedrohungen eingeordnet werden sollen, und wurde von Microsoft eingeführt [24].

- **D**amage Potential - Wie hoch ist der erwartete Schaden, wenn die Schwachstelle ausgenutzt wird?
- **R**eproducibility - Wie einfach ist es, den Angriff zu reproduzieren?
- **E**xploitability - Wie leicht ist es, den Angriff durchzuführen?
- **A**ffected Users - Wie viele Nutzer sind betroffen?
- **D**iscoverability - Wie einfach ist es, die Schwachstelle zu entdecken?

Aufgrund der Tatsache, dass Sicherheit nicht durch Verschleierung erreicht werden sollte, wird der Wert für Discoverability häufig bei der Bewertung ausgeschlossen [vgl. 25]. Auch ansonsten hat das Verfahren in der Praxis einige Schwächen und wurde von Microsoft inzwischen von STRIDE abgelöst, kann aber nach wie vor für eine allgemeine Risikobetrachtung verwendet werden.

6.2.2 STRIDE

Für die Risikoabschätzung ihrer System wird bei Microsoft inzwischen das STRIDE-Modell verwendet, das ein IT-System als ganzes betrachtet und den drei Grundsätzen der IT-Sicherheit Vertraulichkeit, Integrität und Verfügbarkeit sowie den erweiterten Schutzzielen Authentizität und Nichtabstreitbarkeit (Verbindlichkeit) mehr Rechnung trägt. Auch STRIDE wurde nach seinen sechs Kategorien benannt [26].

- **S**poofing - Vorspielung falscher Identitäten, Schutzziel: Authentizität
- **T**ampering - Manipulation von Daten, Schutzziel: Integrität
- **R**epudiation - Abstreitbarkeit, Schutzziel: Verbindlichkeit
- **I**nformation Disclosure - Möglichkeit von Datenabfluss, Schutzziel: Vertraulichkeit

- **D**enial of Service - Nichtverfügbarkeit von Systemressourcen, Schutzziel: Verfügbarkeit
- **E**levation of Privilege - Unberechtigter Zugriff auf Ressourcen, Schutzziel: Authentizität, Verbindlichkeit

STRIDE soll für ein eingesetztes IT-System bewerten, welchen Gefahren dieses ausgesetzt ist oder welches Risiko der Einsatz eines bestimmten IT-Systems birgt. Dies kann nachfolgend in iterativen Prozessen auf das Gesamtsystem ausgeweitet werden, um eine umfassende Risikoanalyse und ein darauf abgestimmtes IT-Sicherheitskonzept ausarbeiten zu können.

Der Nachteil von STRIDE für den Anwendungsfall bei der Betrachtung von Schlüsselaustauschverfahren ist seine Allgemeinheit. So sollen die hier betrachteten Schlüsselaustauschverfahren hauptsächlich Authentizität und Integrität von ausgetauschten Nachrichten sicherstellen. Eine Nichtverfügbarkeit von oder ein unberechtigter Zugriff auf Ressourcen findet bei Angriffen auf Verschlüsselungsverfahren nicht statt. Außerdem bietet die Risikoanalyse nach STRIDE von sich aus nicht die Möglichkeit, einzelne Risiken höher zu gewichten als andere und bietet eher ein Mapping von Angriffsszenarien auf Schutzmaßnahmen.

6.2.3 PASTA

PASTA (Process for Attack Simulation and Threat Analysis) bietet ein prozessbasiertes und risikoorientiertes Verfahren, welches geschäftliche und technische Anforderungen miteinander in Beziehung setzen kann. Der Vorteil hieraus ist, dass eine Risikoanalyse auch in Zusammenarbeit mit der Geschäftsleitung und der Buchhaltung einer Organisation erarbeitet werden kann und bestimmte Risiken als nicht geschäftskritisch oder vertretbar eingestuft werden können. Hierfür schreibt PASTA einen siebenstufigen Prozess vor [27].

1. Define Objectives
2. Define Technical Scope
3. Application Decomposition
4. Threat Analysis
5. Vulnerability & Weaknesses Analysis
6. Attack Modeling

7. Risk & Impact Analysis

Auch PASTA betrachtet ein IT-System oder Geschäftsprozesse als ganzes und kann damit im betrieblichen Umfeld gut eingesetzt werden. Da finanzielle und geschäftliche Prozesse in dieser Arbeit nicht fokussiert betrachtet werden, wäre die Einhaltung des Prozesses nach PASTA zu umfangreich.

6.2.4 Vergleich der Risikoanalyseverfahren

Sowohl STRIDE als auch PASTA werden in der Praxis für Risikoanalysen im betrieblichen Kontext eingesetzt. Vor allem PASTA eignet sich dafür, neben technischen Aspekten auch finanzielle und geschäftliche Prozesse zu betrachten, um so eine umfassende Risikoabschätzung erreichen zu können. Auch STRIDE bietet eine umfangreiche Analysemöglichkeit aller Bedrohungsszenarien eines im betrieblichen Umfeld eingesetzten IT-Systems. Diese allgemeine Anwendbarkeit ist für die hier durchzuführende Analyse jedoch zu umfangreich. Im Rahmen dieser Arbeit existieren keine geschäftlichen Rahmenbedingungen, die zur Überprüfung herangezogen werden können. Die beiden Verfahren müssten daher für die Analyse stark angepasst werden.

Die Risikoanalyse nach DREAD muss für diese Arbeit ebenfalls adaptiert werden, jedoch nicht so stark wie die anderen beiden Verfahren. In den folgenden Abschnitten soll daher das DREAD-Modell nach folgenden Gesichtspunkten verwendet werden.

Die einzelnen betrachteten Analysepunkte werden für jede gefundene Schwäche eines Verfahren betrachtet und mit Punkten von 1 (geringes Risiko) bis 3 (hohes Risiko) bewertet. Als abstraktes Ziel eines Angreifers kann gesehen werden, Informationen über den privaten Schlüssel oder das geteilte Geheimnis von Kommunikationspartnern zu erlangen, um den geheimen Kommunikationskanal zu kompromittieren. Jeder Angriffsvektor wird für die einzelnen Kategorien bewertet und erhält durch Aufsummierung einen Gesamtwert.

- **D**amage Potential: Die Schwäche eines Verfahrens soll danach betrachtet werden, ob das Verfahren selbst eine Lösung für diese Schwäche hat. Kann eine Schwäche beispielsweise durch den Einsatz von mehr Rechen- oder Speicherressourcen behoben werden, wird das Schadenspotenzial als gering (1) eingestuft. Wenn eine Schwachstelle jedoch das gesamte Verfahren angreift, wird das Schadenspotenzial als hoch (3) eingestuft.

- **R**eproducibility: Welche Vorbedingungen müssen erfüllt sein, damit die Schwachstelle ausgenutzt werden kann? Müssen keine Vorbedingungen erfüllt sein, wird die Reproduzierbarkeit als kritisch (3) eingestuft, ansonsten ist die Bewertung je nach Anzahl oder Erfüllbarkeit der Vorbedingungen als mittel (2) oder unkritisch (1) bewertet.
- **E**xploitability: Wie viel Aufwand bedeutet es für einen Angreifer, die Schwachstelle unter den gegebenen Vorbedingungen auszunutzen? Reicht für den Angreifer das Wissen, dass zwischen zwei Personen ein bestimmtes Schlüsselaustauschverfahren verwendet wird, um den Schlüsselaustausch zu kompromittieren, wird dieser Punkt als kritisch (3) bewertet. Sind weitere Informationen über die gewählten Parameter notwendig, wird der Punkt als mittel (2) bewertet. Sind darüber hinaus zusätzliche Informationen notwendig, wird der Punkt als unkritisch (1) bewertet.
- **A**ffected Users: Bei einem gebrochenen Schlüsselaustauschverfahren sind potenziell alle Nutzer des Internets betroffen. Dieser Punkt wird als kritisch (3) bewertet, wenn durch die Kenntnis einer Schwachstelle die komplette Kommunikation mit einem Schlüsselaustauschverfahren als kompromittiert gilt. Schwachstellen, die die Kommunikation einer Person mit allen anderen Personen betrifft werden als mittel (2) eingestuft, betrifft die Schwachstelle nur die Kommunikation einer Person mit genau einer anderen Person, wird dieser Punkt als unkritisch (1) eingestuft.
- **D**iscoverability: Es wird davon ausgegangen, dass Schwachstellen stets sofort gefunden werden und offen verfügbar sind. Dieser Punkt wird daher nicht betrachtet.

Insgesamt sind demnach für jeden Angriffsvektor Risikowerte zwischen 4 und 12 möglich. Die Summe der Kategorien ergibt einen Gesamtwert, der die Kritikalität der Schwachstelle einschätzt. Schwachstellen mit einem Gesamtwert von 11 oder 12 sollen zusätzlich als Ausschlusskriterium gelten, da durch einen Angriffsvektor mit diesem Risikowert die Sicherheit des Verfahrens nicht mehr gegeben sein kann (siehe auch Abb. 6.1).

Wert	Kritikalität der Schwachstelle	Aussage über Sicherheit des Verfahrens
4 - 6	unkritisch	nutzbar mit wenigen zusätzlichen Sicherungsmaßnahmen, Sicherheit bis auf weiteres gegeben
7 - 9	leicht kritisch	nutzbar mit einigen zusätzlichen Sicherungsmaßnahmen, Sicherheit auf beschränkte Zeit gegeben
10	kritisch	nutzbar mit umfangreichen zusätzlichen Sicherungsmaßnahmen oder auf absehbare Zeit unsicher
11	hoch kritisch	Nutzung des Verfahrens nicht empfohlen, Sicherheit vor potenten Akteuren nicht gegeben
12	höchst kritisch	Verfahren ist vollständig gebrochen, jegliche Kommunikation ist als kompromittiert anzusehen

Abb. 6.1: Beziehung der Risikobewertungen zur Sicherheit des Verfahrens

Für die Schlüsselaustauschverfahren sind unterschiedlich viele Schwachstellen bekannt. Über die Gesamtwerte der bewerteten Schwachstellen wird daher ein Mittelwert gebildet, der zum Vergleich der Verfahren herangezogen wird. Sollte eine bekannte Schwachstelle jedoch als „hoch kritisch“ oder „höchst kritisch“ eingestuft sein, stellt dies ein Ausschlusskriterium für die Nutzung des Verfahrens insgesamt dar.

6.3 Diffie-Hellman

Der klassische Diffie-Hellman ist eines der grundlegendsten Verfahren zum Schlüsselaustausch. Nachdem mithilfe des Diffie-Hellman gezeigt wurde, dass sich zwei einander nicht bekannte Parteien mithilfe dieses Verfahrens über einen unsicheren Kommunikationskanal auf ein gemeinsames Geheimnis einigen können, hat das Diffie-Hellman-Verfahren die Grundlage für asymmetrische Schlüsselaustauschverfahren im Internet gelegt.

Das Verfahren seit seiner Entdeckung im Jahr 1976 gut erforscht und anhand vieler Forschung auf seine Sicherheit überprüft worden. Die Sicherheit des DH-Verfahrens basiert dabei auf dem Computational Diffie-Hellman-Problem, das eng verwandt ist mit dem Diskreten-Logarithmus-Problem. Nach aktuellem Stand ist jemand, der in

der Lage ist, den diskreten Logarithmus effizient zu lösen (das heißt in polynomieller Zeit), auch in der Lage ist, das Diffie-Hellman-Problem zu lösen.

6.3.1 Wahl der Primzahl p

Bei der Wahl der Primzahl p muss darauf geachtet werden, dass $p - 1$ mindestens einen großen Primfaktor besitzt. Ist dies nicht der Fall und p besteht ausschließlich aus kleinen Primfaktoren, kann der 1978 entdeckte Algorithmus nach Pohlig-Hellmann angewandt werden, um den diskreten Logarithmus in polynomieller Zeit ($\log^2 p$) zu lösen [28].

Kategorie	**Bewertung**	**Begründung**
Damage Potential	2	Werden ausschließlich kleine Primfaktoren zur Berechnung von $p - 1$ benötigt, kann mithilfe der Pohlig-Hellmann-Methode der geheime Schlüssel $g^a \mod p$ berechnet werden. Die Schlüsselaustausch kann dann als kompromittiert angesehen werden.
Reproducibility	1	Für gängige Diffie-Hellmann-Verfahren gibt es öffentlich bekannte Parameter, die die Anwendbarkeit der Pohlig-Hellmann-Methode ausschließen [21].
Exploitability	2	Ein Angreifer muss Kenntnis über das verwendete Schlüsselaustauschverfahren und über die verwendete Primzahl p besitzen. p ist zwar Bestandteil der öffentlich bekannten Parameter, es kann jedoch nicht von vornherein davon ausgegangen werden, dass ein gewähltes p die Anwendung der Pohlig-Hellmann-Methode ermöglicht.
Affected Users	1	Bei jeder Kommunikation werden neue Schlüssel mit neuen Parametern erzeugt. Es ist daher nur eine 1-zu-1-Kommunikation kompromittiert.
Gesamtwert	6	

Abb. 6.2: Einordnung von Pohlig-Hellmann nach DREAD

6.3.2 Wahl der Basis g

Die Basis g muss außerdem so gewählt werden, dass diese ein Erzeugerelement des gesamten endlichen Körpers $\mathbb{F}_p$ darstellt. Ist dies nicht der Fall, wird die Gruppe

der möglichen Schlüssel $g^{ab} \mod p$ auf eine Untergruppe des Körpers F_p reduziert, deren Ordnung kleiner ist als die des Körpers $\mathbb{F}_p$. Da die Basis g öffentlich bekannt ist, reduziert sich somit die Anzahl der Möglichkeiten auszuprobierender Schlüssel für einen Angreifer dabei auf die Anzahl der Elemente der Untergruppe.

Einen Sonderfall stellt hierbei die Wahl der Basis $g = 1$ dar, da sich hierbei für die erzeugten privaten Schlüssel g^a und g^b jeweils 1 ergibt. Dieser Sonderfall wird hier nicht betrachtet.

Kategorie	**Bewertung**	**Begründung**
Damage Potential	2	Ist die Basis g kein Erzeugerelement des endlichen Körpers $\mathbb{F}_p$, reduziert sich die Anzahl der möglichen Schlüssel auf die Anzahl der aus g generierten Elemente. Dies kann bei sehr großem p dennoch bedeuten, dass das Verfahren angemessen sicher ist, da nach wie vor viele Elemente ausprobiert werden müssen.
Reproducibility	1	Für gängige Diffie-Hellmann-Verfahren gibt es öffentlich bekannte Parameter, die das Erzeugerelement g abhängig von der Primzahl p vorschreiben [21].
Exploitability	2	Ein Angreifer muss Kenntnis über das verwendete Verfahren und die verwendete Basis g haben. g ist zwar Teil der öffentlich bekannten Parameter, es muss jedoch für g berechnet werden, welche Untergruppe von $\mathbb{F}_p$ durch g erzeugt wird.
Affected Users	1	Bei jeder Kommunikation werden neue Schlüssel mit neuen Parametern erzeugt. Es ist daher nur eine 1-zu-1-Kommunikation kompromittiert.
Gesamtwert	6	

Abb. 6.3: Einordnung eines fehlerhaft gewählten Generatorelements nach DREAD

6.3.3 Vorausberechenbarkeit

Die Wahl der öffentlichen Parameter zum Schlüsselaustausch nach DH ergibt jedoch ein neues Problem. Diese öffentlich bekannten Parameter kann ein ausreichend potenter Angreifer nutzen, um die Ergebnisse des Schlüsselaustausches über dem endlichen Körper $\mathbb{F}_p$ vorauszuberechnen. Im Rahmen der Logjam-Schwachstelle im Verschlüsselungsprotokoll Transport Layer Security (TLS) wurde bekannt, dass nach einer Vorberechnung von einer Woche DH-Schlüssel einer Länge von 512-Bit

innerhalb von etwa einer Minute berechnet werden können. Es wird außerdem geschätzt, dass die Vorausberechnung von Schlüsseln für einen 1024-Bit Diffie-Hellman im Rahmen der Möglichkeiten von staatlichen Akteuren liegt [29]. Dies bedeutet für einen sicheren Schlüsselaustausch nach Diffie-Hellman, dass weitaus größere Schlüssellängen notwendig sind, um die potenzielle Angriffsfläche zu minimieren (siehe auch [15]).

Kategorie	**Bewertung**	**Begründung**
Damage Potential	1	Bei der Wahl von höheren Schlüssellängen kann die Vorausberechenbarkeit vermieden werden.
Reproducibility	3	Ein ausreichend potenter Angreifer kann alle erzeugten Schlüssel ohne weitere Vorbedingungen vorberechnen.
Exploitability	2	Der Angreifer muss über große Rechenkapazitäten verfügen. Bei einer Schlüssellänge von 1024 Bit können staatlich finanzierte Akteure alle erzeugbaren Schlüssel vorausberechnen.
Affected Users	3	Für kleine Schlüssellängen können alle möglichen Schlüssel vorausberechnet werden. Es ist daher der Schlüsselaustausch aller Nutzer des Verfahrens betroffen.
Gesamtwert	9	

Abb. 6.4: Einordnung der Vorausberechenbarkeit nach DREAD

6.3.4 Man-in-the-Middle

Ein weiteres Problem des Diffie-Hellman-Schlüsselaustausches ist die Möglichkeit einer Man-in-the-Middle-Attacke. Wenn zwei Parteien A und B sich über einen unsicheren Kommunikationskanal auf ein gemeinsames Geheimnis einigen möchten, könnte eine dritte Partei C sich zwischen die Kommunikation von A und B schalten und mit beiden Parteien jeweils eine eigene Sitzung aufbauen. Da A und B zuvor untereinander nicht bekannt sind, gibt es beim klassischen Diffie-Hellman für beide keine Möglichkeit zu verifizieren, dass sie sich tatsächlich nur miteinander unterhalten. Um einen solchen Angriff zu verhindern, ist zusätzlich eine Authentifizierung der ausgetauschten Nachrichten notwendig, die über ein zusätzliches Protokoll spezifiziert werden muss.

Kategorie	Bewertung	Begründung
Damage Potential	3	Ohne zusätzliche Maßnahmen zur Authentifizierung wird die gesamte Kommunikation kompromittiert.
Reproducibility	1	Die Kommunikationspartner dürfen keine Maßnahmen zur gegenseitigen Authentifizierung einsetzen. Praktisch wird die Authentizität der Kommunikationspartner in der Regel durch überprüfbare Zertifikate sichergestellt.
Exploitability	1	Der Angreifer muss über die Möglichkeit verfügen, sich unbemerkt in die Kommunikation zwischen den beiden Parteien einzuklinken. In einem öffentlichen Netzwerk würde diese Netzmanipulation auffallen, für ein privates Netzwerk müsste der Angreifer sich zuvor Zugang zum lokalen Netz verschafft haben.
Affected Users	1	Jeder Man-in-the-Middle-Angriff betrifft ausschließlich die beiden an der Kommunikation beteiligten Parteien.
Gesamtwert	6	

Abb. 6.5: Einordnung von Man-in-the-Middle nach DREAD

6.3.5 Der Diskrete Logarithmus auf einem Quantencomputer

Das Schlüsselaustauschverfahren nach DH basiert auf dem DLP, das auf klassischen Computern als schwieriges Problem gilt. Dies bedeutet, dass kein Algorithmus existiert, der das Problem in polynomieller Zeit lösen kann. Die Komplexität des Problems nimmt exponentiell zu der Eingabegröße (hier: der Größe des endlichen Körpers $\mathbb{F}_p$) zu und kann so nicht effizient gelöst werden.

1994 wurde von Peter Shor ein Algorithmus entdeckt, der das DLP auf einem praktisch nutzbaren Quantencomputer in polynomieller Zeit lösen kann. Dies bedeutet, dass ein ausreichend potenter Angreifer mit einem praktisch nutzbaren Quantencomputer alle auf dem DLP basierenden Schlüsselaustauschverfahren ohne weitere Vorbedingungen brechen kann [30].

Kategorie	**Bewertung**	**Begründung**
Damage Potential	3	Durch Berechenbarkeit des diskreten Logarithmus können alle erzeugten privaten Schlüssel in polynomieller Zeit berechnet werden.
Reproducibility	3	Die Berechnung des diskreten Logarithmus kann unabhängig von gewählten Parametern und Vorbedingungen durchgeführt werden.
Exploitability	1 bzw. 3	Der Angreifer muss über einen praktisch einsetzbaren Quantencomputer verfügen. Dies ist nach aktuellem Stand nicht möglich, ist aber auf absehbare Zeit wahrscheinlich.
Affected Users	3	Das Brechen des DLP betrifft alle Nutzer eines darauf basierenden Schlüsselaustauschverfahrens.
Gesamtwert	10 bzw. 12	

Abb. 6.6: Einordnung von Shor's Algorithmus nach DREAD

6.3.6 Zusammenfassung

Zusammenfassend lässt sich sagen, dass auf einen klassischen Diffie-Hellmann-Schlüsselaustausch verschiedene Angriffsvektoren existieren, die die Sicherheit des Verfahrens beeinträchtigen. Während einige der Angriffe umgangen werden können, indem öffentlich bekannte Parameter verwendet werden, die diese Schwachstellen vermeiden, können andere Schwachstellen nur durch eine Vergrößerung des zugrundeliegenden endlichen Körpers $\mathbb{F}_p$ durch Wahl eines größeren p oder durch den Einsatz zusätzlicher Protokolle umgangen werden. Die Möglichkeit, das gesamte Verfahren unabhängig der Vorbedingungen mit einem praktisch nutzbaren Quantencomputer zu brechen, stellt jedoch die größte Bedrohung dar.

Angriffsvektor	**Bewertung**
Wahl der Primzahl p	6
Wahl der Basis g	6
Vorausberechenbarkeit	9
Man-in-the-Middle	6
Der Diskrete Logarithmus auf einem Quantencomputer	10 bzw. 12
Mittelwert	7.4 bzw. 7.8

Abb. 6.7: Mittelwert der Angriffsvektoren auf DH nach DREAD

6.4 Elliptic Curve Diffie-Hellman

Aufgrund der Angriffsmöglichkeiten auf das klassische Diffie-Hellman-Verfahren wurden die Schlüssellängen von DH stets erhöht, um ein vergleichbares Sicherheitsniveau beizubehalten. Außerdem wurden die zu verwendenden Parameter veröffentlicht, um Seitenkanalangriffe zu verhindern, die bei ungünstiger Parameterwahl auftreten kann [21].

Um die gewählten Schlüssellängen zu verkürzen, wurde das Diffie-Hellman-Verfahren auf elliptische Kurven auf endlichen Körpern übertragen. Aufgrund der Operationen auf elliptischen Kurven bietet diese weniger Angriffsvektoren und können daher mit geringeren Schlüssellängen ein angemessenes Sicherheitsniveau bereitstellen. Zum jetzigen Zeitpunkt werden in der Praxis hauptsächlich elliptische Kurven verwendet, um einen Schlüsselaustausch zu bewerkstelligen.

Auch bei elliptischen Kurven gibt es inzwischen einige standardisierte Kurven mit öffentlich bekannten Parametern und Kurven. Aufgrund des Einflusses des US-amerikanischen Geheimdienstes National Security Agency (NSA) auf die Standardisierungsverfahren der NIST, stehen jedoch einige der standisierten Kurven im Verdacht, manipulierte Inputs zu enthalten, die eine nachträgliche Berechnung der vereinbarten Schlüssel ermöglichen [31]. Die Berechenbarkeit ist zwar bis heute nicht bewiesen, schwächt aber das Vertrauen in die gewählten Verfahren.

6.4.1 Wahl der Parameter

Ähnlich wie beim klassischen Diffie-Hellmann ist die Sicherheit von ECDH auch zu einem großen Teil von den gewählten Parametern abhängig. Um eine vergleichbare Sicherheit von 128 Bit zu erreichen ist beispielsweise mindestens eine Schlüssellänge von etwa 256 Bit notwendig, somit muss auch der zugrundeliegende endliche Körper $\mathbb{F}_p$ eine Größe von 256 Bit besitzen. Da die Sicherheit von ECDH auf dem ECDLP basiert, das eng verwandt ist mit dem DLP, sind die meisten gängigen Verfahren zur Berechnung des diskreten Logarithmus auch für ECDH anwendbar. Um eine angemessene Sicherheit zu erreichen, ist es daher notwendig, einen ausreichend großen endlichen Körper $\mathbb{F}_p$ bzw. ein ausreichend großes p zu wählen, um die Berechnung des diskreten Logarithmus zu verhindern. Die Berechnung des diskreten Logarithmus ist nach aktuellem Stand am effizientesten mit der Pollard-Rho-Methode möglich. Die Zeitkomplexität des Algorithmus steigt exponentiell mit der Größe der Eingabe p [32].

Kategorie	Bewertung	Begründung
Damage Potential	1	Bei der Wahl von höheren Schlüssellängen kann die Berechnung des diskreten Logarithmus mittels der Pollard-Rho-Methode verhindert werden.
Reproducibility	2	Die Kommunikationspartner müssen sich auf einen kleinen endlichen Körper $\mathbb{F}_p$ geeinigt haben, damit die Pollard-Rho-Methode in absehbarer Zeit ein Ergebnis liefert.
Exploitability	2	Die Berechnung des diskreten Logarithmus nach der Pollard-Rho-Methode wird effizienter, je mehr Systemressourcen dem Angreifer zur Verfügung stehen. Nach aktuellem Stand ist eine Primzahl p der Länge 256 Bit für ECDH ausreichend, um die Berechnung des diskreten Logarithmus in angemessener Zeit zu verhindern.
Affected Users	1	Die Berechnung des diskreten Logarithmus betrifft alle Nutzer eines Verfahrens mit zu geringer Schlüssellänge. Der diskrete Logarithmus muss jedoch für jeden Schlüsselaustausch abhängig von den gewählten Parametern berechnet werden.
Gesamtwert	6	

Abb. 6.8: Einordnung der Pollard-Rho-Methode nach DREAD

6.4.2 Schlüsselvalidierung

Daniel J. Bernstein, Erfinder der Curve25519, argumentiert, dass man bei Benutzung der Curve25519 für einen Diffie-Hellman-Schlüsselaustausch keine Validierung der erzeugten Schlüssel benötigt, da die Kurve so ausgewählt wurde, alle 32-byte Eingaben als valide öffentliche Diffie-Hellman-Schlüssel zu akzeptieren [33].

Demgegenüber entgegnen einige Sicherheitsforscher, dass eine Validierung der öffentlichen in allen Protokollen, die die Curve25519 für einen Diffie-Hellman-Schlüsselaustausch verwenden, notwendig ist, da sonst auch der Punkt in der Unendlichkeit (welcher ebenfalls ein valider Punkt auf der elliptischen Kurve ist) als valider öffentlicher Schlüssel akzeptiert wird (siehe auch [34] und [35]). Dies hätte zur Folge, dass unabhängig von den gewählten (zufällig erzeugten) privaten Schlüsseln das gemeinsame Geheimnis stets die Codierung des Punktes im Unendlichen ergibt. Vor allem bei Protokollen, bei denen ein Kommunikationspartner die Nutzung eines

öffentlichen Schlüssels festlegen kann (sogenannte „non-contributory“, also nicht-gleichberechtigte Protokolle) oder bei unbemerkt an der Kommunikation beteiligten Dritten (siehe auch Kapitel 6.4.3), wäre das geteilte Geheimnis bekannt und die Kommunikation kann als kompromittiert angesehen werden. Das NIST empfiehlt daher die konsequente Validierung aller öffentlichen Schlüssel bei Einsatz eines auf ECC basierenden Protokolls [36, vgl. Kapitel 5.6.2.3.2].

Aufgrund von Angriffen auf einige ECDH-Implementierungen des TLS-Standards wird eine Überprüfung der erzeugten Punkte der elliptischen Kurve auch bei gleichberechtigten Protokollen empfohlen [37].

Kategorie	**Bewertung**	**Begründung**
Damage Potential	3	Eine fehlende Validierung der öffentlichen Schlüssel oder des daraus erzeugten gemeinsamen Schlüssels kann nicht durch ECDH behoben werden, solange sich die erzeugten Werte als valide Punkte der zugrundeliegenden elliptischen Kurve ergeben. Stattdessen muss eine Validierung auf höherer Protokollebene geschehen.
Reproducibility	1	Das zugrundeliegende Kommunikationsprotokoll muss ein nicht-gleichberechtigtes Protokoll sein oder nicht-authentifizierte Kommunikation ermöglichen (siehe auch Kapitel 6.4.3). Das NIST empfiehlt die Validierung öffentlicher Schlüssel.
Exploitability	1	Ein Angreifer muss Zugriff auf die Schlüsselerzeugung eines höherberechtigten Protokollteilnehmers oder auf den Schlüsselaustausch beider gleichberechtigter Protokollteilnehmer besitzen.
Affected Users	2	Je nach eingesetztem Kommunikationsprotokoll betrifft die Schwachstelle die komplette Kommunikation eines höherberechtigten Protokollteilnehmers oder eine 1-zu-1-Kommunikation.
Gesamtwert	7	

Abb. 6.9: Einordnung der ECDH-Schlüsselvalidierung nach DREAD

6.4.3 Man-in-the-Middle

Ähnlich wie bei einem klassischen DH-Schlüsselaustausch bietet ECDH keine Authentizitätsprüfung der teilnehmenden Kommunikationspartner. Somit ist es für einen Angreifer ebenfalls möglich, sich unbemerkt zwischen beide Kommunikationsparteien zu schalten und mit beiden jeweils einen eigenen Schlüsselaustausch zu initiieren. Analog zu DH muss die Authentizität der Kommunikationsparteien durch ein zusätzliches Protokoll sichergestellt werden.

Kategorie	**Bewertung**	**Begründung**
Damage Potential	3	Ohne zusätzliche Maßnahmen zur Authentifizierung wird die gesamte Kommunikation kompromittiert.
Reproducibility	1	Die Kommunikationspartner dürfen keine Maßnahmen zur gegenseitigen Authentifizierung einsetzen. Praktisch wird die Authentizität der Kommunikationspartner in der Regel durch überprüfbare Zertifikate sichergestellt.
Exploitability	1	Der Angreifer muss über die Möglichkeit verfügen, sich unbemerkt in die Kommunikation zwischen den beiden Parteien einzuklinken. In einem öffentlichen Netzwerk würde diese Netzmanipulation auffallen, für ein privates Netzwerk müsste der Angreifer sich zuvor Zugang zum lokalen Netz verschafft haben.
Affected Users	1	Jeder Man-in-the-Middle-Angriff betrifft ausschließlich die beiden an der Kommunikation beteiligten Parteien.
Gesamtwert	6	

Abb. 6.10: Einordnung von Man-in-the-Middle nach DREAD

6.4.4 Der Diskrete Logarithmus auf einem Quantencomputer

Da das ECDH zugrundeliegende ECDLP eng verwandt ist mit dem klassischen DLP, ist ECDH ebenfalls anfällig für Shor's Algorithmus auf einem praktisch nutzbaren Quantencomputer. Genauso wie der klassische DH-Schlüsselaustausch gelten auch alle ECDH-Schlüsselaustauschverfahren mit dem Auftreten von praktisch nutzbaren Quantencomputern als unsicher [38].

Kategorie	Bewertung	Begründung
Damage Potential	3	Durch Berechenbarkeit des diskreten Logarithmus können alle erzeugten privaten Schlüssel in polynomieller Zeit berechnet werden.
Reproducibility	3	Die Berechnung des diskreten Logarithmus kann unabhängig von gewählten Parametern und Vorbedingungen durchgeführt werden.
Exploitability	1 bzw. 3	Der Angreifer muss über einen praktisch einsetzbaren Quantencomputer verfügen. Dies ist nach aktuellem Stand nicht möglich, ist aber auf absehbare Zeit wahrscheinlich.
Affected Users	3	Das Brechen des ECDLP betrifft alle Nutzer eines darauf basierenden Schlüsselaustauschverfahrens.
Gesamtwert	10 bzw. 12	

Abb. 6.11: Einordnung von Shor's Algorithmus nach DREAD

6.4.5 Zusammenfassung

Es lässt sich erkennen, dass ECDH ähnliche Angriffsvektoren besitzt wie ein klassisches DH-Verfahren. Die Größe des gewählten endlichen Körpers $\mathbb{F}_p$ spielt eine große Rolle, um klassische Faktorisierungsangriffe zu verhindern. Wie bei DH sind jedoch zusätzliche Protokolle zur Validierung der Eingaben und zur Sicherstellung einer 1-zu-1-Kommunikation notwendig. Außerdem stehen einige öffentliche elliptische Kurven im Verdacht, versteckte Hintertüren zu enthalten. Dies ist jedoch nicht bewiesen.

Angriffsvektor	Bewertung
Wahl der Parameter	6
Schlüsselvalidierung	7
Man-in-the-Middle	6
Berechnung des Diskreten Logarithmus auf einem Quantencomputer	10 bzw. 12
Mittelwert	7.25 bzw. 7.75

Abb. 6.12: Mittelwert der Angriffsvektoren auf ECDH nach DREAD

6.5 Supersingular Isogeny Diffie-Hellmann (SIDH)

SIDH ist das erste isogeniebasierte Schlüsselaustauschverfahren, das ein klassisches Diffie-Hellman-Verfahren ersetzen kann. Für die beiden hier betrachteten isogeniebasierten Verfahren gilt jedoch, dass diese sehr junge Verfahren darstellen und weiterer eingehender Forschung benötigen.

Im Gegensatz zu klassischen Schlüsselaustauschverfahren gründet sich die Sicherheit isogeniebasierter Schlüsselaustauschverfahren nicht auf der mathematischen Schwierigkeit des DLP, sondern auf der Schwierigkeit, bei zwei gegebenen elliptische Kurven einen Homomorphismus zwischen beiden zu finden. Allgemein formuliert berechnen sich die beiden bei einem Schlüsselaustausch beteiligten Parteien genau einen Isogeniepfad zwischen der Ausgangskurve und der Ergebniskurve, welche dann als gemeinsames Geheimnis verwendet wird. Ein Angreifer hat lediglich die Möglichkeit, alle möglichen Isogenien der Ausgangskurve zu berechnen, was bei großen endlichen Körpern nicht effizient lösbar ist.

6.5.1 Wahl der Primzahl p

Bei SIDH wird üblicherweise eine zusammengesetzte Primzahl der Form $p = 2^{e_a}3^{e_b} - 1$ verwendet. Dabei muss darauf geachtet werden, dass gilt $2^{e_a} \approx 3^{e_b}$. Ist dies nicht der Fall, ist die Anzahl der möglichen Abbildungskerne für einen der beiden an der Kommunikation beteiligten Partner ungleich kleiner. Damit würde sich die Anzahl der möglichen Untergruppen zur Erzeugung des eigenen öffentlichen Schlüssels verringern, wodurch ein Ungleichgewicht in der Sicherheit des Protokolls entsteht.

Kategorie	Bewertung	Begründung
Damage Potential	1	Wenn $2^{e_a} \approx 3^{e_b}$ nicht gilt, verringert sich die Anzahl der möglichen öffentlichen Schlüssel einer der beiden Parteien auf den jeweils kleineren Wert. Dies kann bei ausreichend großen Werten dennoch eine angemessene Sicherheit bieten.
Reproducibility	1	Die Spezifikation von SIDH schlägt öffentliche Parameter für 2^{e_a} und 3^{e_b} vor, die die Bedingung $2^{e_a} \approx 3^{e_b}$ erfüllen. [17, s. Kapitel 1.6]
Exploitability	1	Ein Angreifer muss Kenntnis über das verwendete Verfahren und die verwendeten Parameter 2^{e_a} und 3^{e_b} haben. Sind diese nicht veröffentlicht, ist nur $p = 2^{e_a}3^{e_b} - 1$ Teil der öffentlich bekannten Parameter. Ein Angreifer müsste p zunächst faktorisieren, um entscheiden zu können, ob diese beiden Untergruppen nicht die Bedingung $2^{e_a} \approx 3^{e_b}$ erfüllen.
Affected Users	2	Bei jeder Kommunikation werden neue Schlüssel mit neuen Parametern erzeugt. Es ist daher nur eine 1-zu-1-Kommunikation kompromittiert.
Gesamtwert	5	

Abb. 6.13: Einordnung einer fehlerhaft gewählten SIDH-Primzahl nach DREAD

6.5.2 Wiederverwendung von Schlüsseln

Ein reines SIDH-Verfahren besitzt den Nachteil, dass das erzeugte Schlüsselpaar nur für eine einzigen Schlüsselaustausch verwendet werden darf. Dies resultiert daraus, dass SIDH keine Überprüfung enthält, ob der Kommunikationspartner einen manipulierten öffentlichen Schlüssel verwendet. [3, s. Kapitel 7]. Ein Angreifer könnte bei mehrfach verwendeten Schlüsseln die Kommunikation zwischen zwei Parteien mithören und einen entsprechenden manipulierten öffentlichen Schlüssel nutzen, um mit einer der beiden Parteien einen weiteren Schlüsselaustausch zu initiieren und somit Informationen über den ersten geheimen Schlüssel zu erfahren. Die Berechnung des entsprechenden gemeinsamen Schlüssels aus dem manipulierten Schlüssel ist zwar eine unzulässige Ausführung des Protokolls, das Schlüsselaustauschverfahren erzwingt jedoch trotzdem die Berechnung des gemeinsamen Schlüssel [39, s. Kapitel 5.2]. Diese Schwachstelle kann nur verhindert werden, wenn jeder Schlüsselaustausch die Erzeugung eines neuen Schlüssels voraussetzt oder ein zusätzliches

Protokoll eingesetzt wird, um Langzeitschlüssel zu verwenden und andere öffentliche Schlüssel auf ihre Gültigkeit zu überprüfen.

Kategorie	**Bewertung**	**Begründung**
Damage Potential	2	Werden statische Schlüssel ohne entsprechende Validierung verwendet, kann Information über das gemeinsame Geheimnis preisgegeben werden. Zur Vermeidung können mit SIDH erzeugte Schlüssel ausschließlich flüchtig eingesetzt werden. In der Praxis werden meist Schlüsselaustauschverfahren mit zusätzlichen Maßnahmen zur Validierung öffentlicher Schlüssel eingesetzt.
Reproducibility	2	Ein Angreifer muss einen zuvor erfolgten Schlüsselaustausch mitgehört haben und die entsprechende Information besitzen, dass eine der beiden Parteien statische Schlüssel ohne Validierung verwendet.
Exploitability	2	Nach einem mitgehörtem Schlüsselaustausch zweier Parteien kann ein Angreifer versuchen, aus den hieraus bekannten Parametern einen manipulierten Schlüsselaustausch zu initiieren.
Affected Users	2	Es sind alle Kommunikationen einer Partei, die statische Schlüssel ohne Validierung verwendet, betroffen.
Gesamtwert	8	

Abb. 6.14: Einordnung der Wiederverwendung von SIDH-Schlüsseln nach DREAD

6.5.3 Meet-in-the-Middle

Der bisher effizienteste Weg zur Berechnung einer unbekannten Isogenie der Ordnung l^e zwischen zwei bekannten Kurven E_0 und E' ist der sogenannte Meet-in-the-Middle-Angriff. In der Komplexitätstheorie ist dieser Angriff eine Umsetzung des Claw-Finding-Algorithmus und ist damit eng verwandt mit dem Finden von Kollisionen von Hashing-Verfahren.

Um die Isogenie $E_0 \mapsto E'$ der Ordnung l^e zu berechnen, ohne die hierfür benötigten Punkte zu kennen, berechnet ein Angreifer alle möglichen Isogenien der Ordnung $l^{\frac{e}{2}}$ der Ausgangskurve E_0 und alle möglichen Isogenien der Ordnung $l^{\frac{e}{2}}$ der Ergebniskurve E'. Die so berechneten Isogenien werden mit hoher Wahrscheinlichkeit eine gemeinsame Zielkurve E'' besitzen, aus der sich durch Komposition $E_0 \mapsto E'' \circ E'' \mapsto E'$

die unbekannte Isogenie $E_0 \mapsto E'$ erzeugen lässt. Der Angriff läuft auf einem klassischen Computer in Abhängigkeit vom gewählten p in einer Komplexität von $O(\sqrt[4]{p})$ [40]. Der gleiche Angriff kann auf einen Quantencomputer übertragen werden und läuft dort mit einer Zeitkomplexität von $O(\sqrt[6]{p})$ [17, s. Kapitel 4.1].

Das Meet-in-the-Middle-Verfahren stellt zwar einen Angriff auf die Parameter von SIDH dar, jedoch nicht direkt auf SIDH selbst, da das gemeinsame errechnete Geheimnis (also die Ergebniskurve bzw. die j-Invarianten von isomorphen Kurven) nicht öffentlich ist. Ein Angreifer erhält somit ausschließlich Informationen über die verwendeten Isogenien der beiden beteiligten Parteien zur Berechnung ihrer jeweiligen öffentlichen Schlüssel.

Kategorie	**Bewertung**	**Begründung**
Damage Potential	1	Die unbekannte Isogenie $E_0 \mapsto E'$ kann mithilfe des Meet-in-the-middle-Verfahrens in polynomieller Zeit gefunden werden. Die Wahl einer entsprechend großen Primzahl p erschwert das Verfahren. Es existiert bisher noch keine praktische Umsetzung zu Berechnung des gemeinsamen Geheimnisses durch das Meet-in-the-Middle-Verfahren.
Reproducibility	3	Das Meet-in-the-Middle-Verfahren ist unabhängig von weiteren Vorbedingungen.
Exploitability	2	Die Berechnung einer unbekannten Isogenie $E_0 \mapsto E'$ ist zwar in polynomieller Zeit möglich, hat aber einen hohen Speicherbedarf. [41, s. Kapitel 6]
Affected Users	2	Das Meet-in-the-Middle-Verfahren betrifft alle Nutzer von SIDH, muss aber für jeden Schlüsselaustausch einzeln berechnet werden. Da SIDH-Schlüssel nicht wiederverwendet werden, betrifft der Angriff lediglich einen 1-zu-1-Schlüsselaustausch.
Gesamtwert	8	

Abb. 6.15: Einordnung des Meet-in-the-Middle-Verfahrens nach DREAD

6.5.4 Zusammenfassung

Die isogeniebasierte Kryptographie ist ein sehr junges Feld und wird stetig weiterentwickelt und erforscht. Die mathematischen Grundlagen sind zwar bekannt, werden aber erst seit kurzem im kryptographischen Kontext verwendet und unter diesem

Aspekt auf ihre Sicherheit überprüft. Es existieren daher noch wenige gesicherte Erkenntnisse über die Zuverlässigkeit dieser Verfahren. Dennoch gibt es bereits einige potenzielle Angriffsvektoren auf die Struktur von SIDH, die in Zukunft dazu führen könnten, dass das Verfahren als unsicher eingestuft wird.

Angriffsvektor	**Bewertung**
Wahl der Primzahl p	5
Wiederverwendung von Schlüsseln	8
Meet-in-the-Middle	8
Mittelwert	7

Abb. 6.16: Mittelwert der Angriffsvektoren auf SIDH nach DREAD

6.6 Commutative Supersingular Isogeny Diffie-Hellmann (CSIDH)

Das isogeniebasierte Schlüsselaustauschverfahren CSIDH hat viele Ähnlichkeiten mit SIDH, bietet jedoch weitaus kürzere Schlüssel und eine integrierte Schlüsselvalidierung. Aufgrund ähnlicher zugrundeliegender Verfahren besitzt CSIDH daher ähnliche Angriffsvektoren.

6.6.1 Wahl der Primzahl p

Bei CSIDH werden private Schlüssel auf andere Weise erzeugt, weshalb die Zusammensetzung der Primzahl p eine weniger gravierende Rolle spielt. Bei CSIDH bestehen private Schlüssel aus einer Folge von Ganzzahlen, die mit den Primzahlfaktoren faktorisiert werden [18, s. Kapitel 6]. Die Primzahl p wird berechnet aus einer bestimmten Anzahl kleiner Primzahlen $p = 4 \cdot l_1 \cdots l_n - 1$ berechnet. Beide am Schlüsselaustausch beteiligten Parteien nutzen die gleichen Faktoren $(l_1, ..., l_n)$, weshalb kein Ungleichgewicht zwischen beiden Parteien entstehen kann.

6.6.2 Hidden-shift-Problem

Das sogenannten Hidden-shift-Problem besagt, dass es zu einer gegebenen Abbildung f eine weitere Abbildung g gibt, für die gilt $f(x) = g(x + s)$. Dabei gilt es, s zu bestimmen, um eine zweite, von f unabhängige Funktion zu erhalten, die auf x die gleiche Abbildung durchführt. Für isogeniebasierte Verfahren würde dies bedeuten, dass für die Isogenie $\phi(E_0)$ eine unabhängige Abbildung $\psi(E_0 + s)$ existiert, die zum gleichen Ergebnis führt. Es kann gezeigt werden, dass für die Berechnung von s

auf Quantencomputern ein Algorithmus existiert, der in subexponentieller Zeit läuft [42].

Kategorie	**Bewertung**	**Begründung**
Damage Potential	2	Das Hidden-shift-Problem gibt eine Möglichkeit, die unbekannte Isogenie auf einem anderen Weg zu berechnen als den Schlüsselaustausch selbst durchzuführen. Da der Algorithmus jedoch subexponentiell (nicht polynomiell) skaliert, kann die Berechnung mit der Wahl eines größeren endlichen Körpers $\mathbb{F}_p$ stark erschwert werden.
Reproducibility	2	Das Meet-in-the-Middle-Verfahren ist abhängig von der Kenntnis der Ausgangskurve E_0 und des zugrundeliegenden endlichen Körpers $\mathbb{F}_p$.
Exploitability	1	Die Lösung des Hidden-shift-Problems benötigt einen praktisch nutzbaren Quantencomputer und hat subexponentiell wachsenden Speicherbedarf.
Affected Users	1	Die Lösung des Hidden-shift-Problems betrifft jeweils nur eine erzeugte Isogenie bei einem 1-zu-1-Schlüsselaustausch.
Gesamtwert	6	

Abb. 6.17: Einordnung des Hidden-shift-Problems nach DREAD

6.6.3 Meet-in-the-Middle

Ähnlich wie bei SIDH kann ein Angreifer bei CSIDH alle Isogenien der Ausgangskurve E_0 und der Ergebniskurve E' einer bestimmten Ordnung berechnen, um dadurch eine gemeinsamen Isogeniepfad über E'' zu erhalten [18, s. Kapitel 7.1]. Das Verfahren hat auch bei CSIDH die gleiche Kritikalität.

Kategorie	Bewertung	Begründung
Damage Potential	1	Die unbekannte Isogenie $E_0 \mapsto E'$ kann mithilfe des Meet-in-the-Middle-Verfahrens in polynomieller Zeit gefunden werden.
Reproducibility	3	Das Meet-in-the-Middle-Verfahren ist unabhängig von weiteren Vorbedingungen.
Exploitability	2	Die Berechnung einer unbekannten Isogenie $E_0 \mapsto E'$ ist in polynomieller Zeit möglich [41, s. Kapitel 6].
Affected Users	2	Das Meet-in-the-Middle-Verfahren betrifft alle Nutzer von CSIDH, muss aber für jeden Schlüsselaustausch einzeln berechnet werden.
Gesamtwert	8	

Abb. 6.18: Einordnung des Meet-in-the-Middle-Verfahrens nach DREAD

6.6.4 Zusammenfassung

Das kommutative isogeniebasierte Schlüsselaustauschverfahren CSIDH ist aufgrund seiner noch sehr jungen Geschichte eines der am wenigsten betrachtete Verfahren. Es konnte außerdem nicht zum Standardisierungsverfahren des NIST eingereicht werden und wird daher nicht geprüft, weshalb bisher wenige Angriffe bekannt sind. Auf einige bekannte Angriffe wird in der Spezifikation von CSIDH eingegangen, bedürfen jedoch noch weiterer Untersuchung [18, s. Kapitel 7].

Angriffsvektor	Bewertung
Hidden-shift-Problem	6
Meet-in-the-Middle	8
Mittelwert	7

Abb. 6.19: Mittelwert der Angriffsvektoren auf CSIDH nach DREAD

6.7 Vergleich

Im Vergleich der Risikoanalysen der verschiedenen Modelle zeigt sich, dass alle Schlüsselaustauschverfahren verschiedenen Angriffsvektoren ausgesetzt sind. Bei allen Verfahren liegen mathematisch schwere Probleme zugrunde, die immer neu untersucht und bewertet werden. Es ist daher stets zu empfehlen, kryptographische Verfahren nicht selbst zu entwickeln, sondern auf öffentlich verfügbare und getestete Verfahren zurückzugreifen. Die erst vor kurzer Zeit entwickelten Verfahren in der isogeniebasierten Kryptographie scheinen aktuell eine angemessene Sicherheit zu

bieten, ein praktischer Einsatz wird dennoch nicht empfohlen, da diese noch nicht ausreichend untersucht sind. In den Untersuchungen des NIST steht ein isogeniebasiertes Schlüsselaustauschverfahren zur Diskussion, die klassischen Verfahren nach der Entwicklung eines praktisch einsetzbaren Quantencomputers zu ersetzen. Währenddessen werden die isogeniebasierten Verfahren weiter verbessert und effizienter gestaltet.

Während bei der Risikoanalyse wurde bei den klassischen Schlüsselaustauschverfahren betrachtet, dass ein praktisch einsetzbarer Quantencomputer die den Verfahren zugrundeliegenden mathematischen Probleme lösen kann. In den Mittelwerten wurde diese Betrachtung durch zwei verschiedene Gesamtwerte berücksichtigt. Sobald ein Quantencomputer die klassische Kryptographie brechen kann, sind jedoch diese Verfahren als unsicher zu betrachten und nicht mehr zu benutzen.

Verfahren	**Mittelwert der Risikoanalyse**
Diffie-Hellman	7.4 bzw. 7.8
Elliptic Curve Diffie-Hellman	7.25 bzw. 7.75
Supersingular Isogeny Diffie-Hellman	7.25
Commutative Supersingular Isogeny Diffie-Hellman	7

Abb. 6.20: Vergleich der Risikowerte der verschiedenen Verfahren nach DREAD

7 Fazit

7.1 Zusammenfassung

In der vorliegenden Arbeit wurden isogeniebasierte Diffie-Hellman-Schlüsselaustauschverfahren mit den klassischen Diffie-Hellman-Verfahren verglichen und auf ihre praktische Einsetzbarkeit untersucht. Die internationale Kryptographie-Community sucht nach einem Weg, eine sichere Kommunikation auch nach der Entwicklung von Quantencomputern zu ermöglichen. Das NIST hat hierzu eine Ausschreibung für Post-Quanten-Kryptographie gestartet, bei welcher auch ein isogeniebasiertes Verfahren in der aktuell dritten Runde vertreten ist. Die Standardisierung der Post-Quanten-Kryptographie ist ein wichtiger Schritt, Schlüsselaustauschverfahren auf die Post-Quanten-Ära zu übertragen.

In dieser Arbeit wurden isogeniebasierte Schlüsselaustauschverfahren als Teilgebiet der Post-Quanten-Kryptographie vorgestellt. Diese wurden mit klassischen Diffie-Hellman-Verfahren auf ihre Zeitkomplexität auf einen klassischen Computer untersucht und verglichen. Es zeigt sich, dass die bisher eingesetzten klassischen Verfahren weitaus optimierter sind und daher ein besseres Zeitverhalten aufweisen können als isogeniebasierte Verfahren. Dies ist nicht zuletzt auch der kryptographischen Forschung der letzten Jahrzehnte zu verdanken. Dennoch kann gesagt werden, dass praktisch nutzbare, isogeniebasierte Schlüsselaustauschverfahren existieren, die die klassischen Verfahren ersetzen können. Hierbei müssen weitere zukünftige Entwicklungen abgewartet werden, die die isogeniebasierten Verfahren effizienter gestalten können.

In einem weiteren Schritt wurden bisher bekannte Angriffe auf Schlüsselaustauschverfahren ausgewertet und in einer Risikoanalyse miteinander verglichen. Es zeigt sich, dass die isogeniebasierten Verfahren auch in der Sicherheit mit den klassischen Verfahren auf einer Höhe sind. Vor allem an dieser Stelle ist die Analyse jedoch mit Vorsicht zu betrachten, da die kryptographische Forschung in diesem jungen Gebiet noch nicht sehr weit fortgeschritten ist und einer langfristigeren Betrachtung bedarf. Es kann daher nicht mit Gewissheit gesagt werden, dass in den nächsten Jahren kein

Angriff auf isogeniebasierte Verfahren entdeckt wird, der diese für kryptographische Bedarfe ausschließt.

Abschließend kann jedoch bemerkt werden, dass isogeniebasierte Verfahren bis zuletzt aussichtsreiche Kandidaten für Post-Quanten-Kryptographie sind. Vor allem das sehr junge CSIDH-Verfahren kann aufgrund der kommutativen Struktur und sehr kurzer Schlüssellängen als Ersatz für aktuelle Schlüsselaustauschverfahren dienen, die auch bei Geräten mit begrenztem Speicherplatz eingesetzt werden.

7.2 Ausblick

7.2.1 Informationsabfluss durch veröffentlichte Punkte in SIDH

Um den SIDH-Schlüsselaustausch zu ermöglichen, müssen beide am Schlüsselaustausch beteiligten Parteien Punkte der jeweils anderen Partei auf dem eigenen erzeugten Isogeniepfad übertragen und die Abbildung der Punkte an die andere Partei kommunizieren. Obwohl bisher keine Angriff auf diese zusätzlich veröffentlichten Informationen bekannt ist, werden bereits Algorithmen erforscht, die potenziell die Grundlage eines praktischen Angriffes bilden könnten [43].

Auch wenn der aktuell vielversprechendste Angriff auf SIDH das Meet-in-the-Middle-Verfahren ist, kann nicht ausgeschlossen werden, dass in Zukunft ein praktischer Angriff veröffentlicht wird, der die zusätzliche Information über die Punktabbildungen verwendet.

7.2.2 Laufzeit von CSIDH

Das CSIDH-Verfahren zeichnet sich durch kurze Schlüssellängen aus, hat jedoch eine ungleich größere Laufzeit. Diese Laufzeit kann durch verschiedene Verbesserung in der Software schon jetzt um den Faktor 1,3 verkürzt werden [44]. Es ist davon auszugehen, dass diese Effizienzsteigerung weiter vorangetrieben wird, um die Praktikabilität von CSIDH weiter zu erhöhen.

7.2.3 Quantum-Sicherheitsanalyse von CSIDH

2018 haben Xavier Bonnetain und André Schrottenloher eine Quantum-Analyse der CSIDH-Parameter veröffentlicht. In ihrer Analyse legen die Sicherheitsforscher nahe, dass die gewählten Parameter nicht das angegebene Sicherheitsniveau erreichen

würden. So wurde CSIDH auf einem 512-Bit Körper equivalent zur Sicherheit von AES-128 angegeben, Bonnetain und Schrottenloher zeigen jedoch, dass für dieses Sicherheitsniveau ein endlicher Körper der Größe von mindestens 1024 Bit notwendig ist [45].

Das Team hinter CSIDH widerspricht dieser Darlegung jedoch und verweist auf ihre eigene Analyse [46, z.B. Überschrift 1.3].

Die Diskussion zeigt jedoch, dass die Sicherheitsanalyse von CSIDH noch nicht abgeschlossen ist und die Forschung am Verfahren noch andauert. Vor dem Hintergrund der langjährigen Analysen und Forschungen bei klassischen Schlüsselaustauschverfahren kann die Sicherheit von CSIDH daher noch nicht endgültig bewertet werden.

Literaturverzeichnis

[1] Bosch, Siegfried: *Algebra*. Mai 2020. – ISBN 978–3–662–61648–2

[2] Feo, Luca de: *Mathematics of Isogeny Based Cryptography*. Mai 2017. – https://arxiv.org/pdf/1711.04062v1.pdf

[3] Costello, Craig: Supersingular Isogeny Key Exchange for Beginners. In: *Selected Areas in Cryptography – SAC 2019* (Jan 2020), S. 21–50

[4] NIST: *Post-Quantum Cryptography PQC*. https://csrc.nist.gov/projects/post-quantum-cryptography, Jan 2017. – [Online; zuletzt aufgerufen am 21.09.2021]

[5] Bernstein, Daniel J. ; Lange, Tanja: *Post-quantum cryptography - dealing with the fallout of physics success*. Cryptology ePrint Archive, Report 2017/314, Apr 2017. – https://ia.cr/2017/314

[6] Ritz, Jennifer: *Masterthesis: Post-Quantum Kryptographie - Evaluation Quantencomputer-resistenter Public-Key-Verfahren basierend auf der Implementierung auf klassischen Computern*, Hochschule Wismar, Diplomarbeit, Jul 2020

[7] Merkle, Ralph C.: Secure Communications Over Insecure Channels. In: *Communications of the ACM* (Apr 1978), S. 294–299

[8] Diffie, W. ; Hellman, M. E.: New Directions in Cryptography. In: *IEEE Transactions on Information Theory* (Nov 1976), S. 644–654

[9] Koblitz, Neil: Elliptic Curve Cryptosystems. In: *Mathematics of Computation* (Jan 1985), S. 203–209

[10] Crypto++: *Crypto++ Library 8.5*. https://www.cryptopp.com/, Apr 2014. – [Online; zuletzt aufgerufen am 03.07.2021]

[11] PQCrypto, Microsoft: *SIDH v3.4 (C Edition)*. https://github.com/Microsoft/PQCrypto-SIDH, Apr 2017. – [Online; zuletzt aufgerufen am 03.07.2021]

[12] Castryck, Wouter ; Lange, Tanja ; Martindale, Chloe ; Panny, Lorenz ; Renes, Joost: *CSIDH: An efficient post-quantum commutative group action.* https://csidh.isogeny.org/software.html, Okt 2018. – [Online; zuletzt aufgerufen am 03.07.2021]

[13] Barker, Elaine: *Recommendation for Key Management: Part 1 - General.* https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf, Mai 2020

[14] Giry, Damien: *BlueKrypt Cryptographic Key Length Recommendation.* https://www.keylength.com/en/compare/, Mai 2020. – [Online; zuletzt aufgerufen am 27.07.2021]

[15] IETF: *Determining Strengths For Public Keys Used For Exchanging Symmetric Keys.* https://www.ietf.org/rfc/rfc3766.txt, Apr 2004

[16] IETF: *Elliptic Curves for Security.* https://www.ietf.org/rfc/rfc7748.txt, Jan 2016

[17] Jao, David ; Azarderakhsh, Reza ; Campagna, Matthew ; Costello, Craig ; Feo, Luca ; Hess, Basil ; Jalali, Amir ; Koziel, Brian ; LaMacchia, Brian ; Longa, Patrick ; Naehrig, Michael ; Pereira, Geovandro ; Renes, Joost ; Soukharev, Vladimir ; Urbanik, David: *Supersingular Isogeny Key Encapsulation.* https://sike.org/files/SIDH-spec.pdf, Apr 2019

[18] Castryck, Wouter ; Lange, Tanja ; Martindale, Chloe ; Panny, Lorenz ; Renes, Joost: *CSIDH: An Efficient Post-Quantum Commutative Group Action.* https://csidh.isogeny.org/csidh-20181118.pdf, Nov 2018

[19] Crypto++: *DH Struct Reference Crypto++ 8.5.* https://cryptopp.com/docs/ref/struct_d_h.html, Jul 2021. – [Online; zuletzt aufgerufen am 27.07.2021]

[20] Crypto++: *Crypto++ Wiki: Diffie-Hellman.* https://www.cryptopp.com/wiki/Diffie-Hellman, Apr 2021. – [Online; zuletzt aufgerufen am 27.07.2021]

[21] IETF: *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE).* https://www.ietf.org/rfc/rfc3526.txt, Mai 2003

[22] Crypto++: *x25519 Class Reference Crypto++ 8.5.* https://cryptopp.com/docs/ref/classx25519.html, Jul 2021. – [Online; zuletzt aufgerufen am 27.07.2021]

[23] Bernstein, Daniel J.: *Curve25519: new Diffie-Hellman speed records*. https://cr.yp.to/ecdh/curve25519-20060209.pdf, Feb 2006

[24] Microsoft: *Threat Modeling*. https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff648644(v=pandp.10), Jun 2003. – [Online; zuletzt aufgerufen am 01.09.2021]

[25] Open Stack: *OpenStack Security Advisories: Metrics*. https://wiki.openstack.org/wiki/Security/OSSA-Metrics#Calibration, 2013. – [Online; zuletzt aufgerufen am 01.09.2021]

[26] Kohnfelder, Loren ; Garg, Praerit: *The threats to our products*. https://adam.shostack.org/microsoft/The-Threats-To-Our-Products.docx, Apr 1999

[27] UcedaVelez, Tony: *Application Threat Modeling via the PASTA Methodology*. https://owasp.org/www-pdf-archive/APAC13_TonyUV.pdf, 2013

[28] Pohlig, Stephen ; Hellmann, Martin: *An Improved Algorithm for Computing Logarithms over GP(p) and Its Cryptographic Significance*. https://ee.stanford.edu/~hellman/publications/28.pdf, Jan 1978

[29] Adrian, David ; Bhargavan, Karthikeyan ; Durumeric, Zakir ; Gaudry, Pierrick ; Green, Matthew ; Halderman, J. A. ; Heninger, Nadia ; Springall, Drew ; Thomé, Emmanuel ; Valenta, Luke ; VanderSloot, Benjamin ; Wustrow, Eric ; Zanella-Béguelin, Santiago ; Zimmermann, Paul: Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, Association for Computing Machinery, 2015 (CCS '15). – ISBN 9781450338325, S. 5–17

[30] Shor, Peter W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. In: *SIAM Journal on Computing* 26 (Okt 1997), Nr. 5, 1484–1509. http://dx.doi.org/10.1137/s0097539795293172. – DOI 10.1137/s0097539795293172. – ISSN 1095–7111

[31] Bernstein, Daniel J. ; Chou, Tung ; Chuengsatiansup, Chitchanok ; Hülsing, Andreas ; Lange, Tanja ; Niederhagen, Ruben ; Vredendaal, Christine van: *How to manipulate curve standards: a white paper for the black hat*. Cryptology ePrint Archive, Report 2014/571, Jul 2014. – https://ia.cr/2014/571

[32] Pollard, John M.: Monte Carlo Methods for Index Computation mod p. In: *Mathematics of Computation* 32 (1978), Nr. 143, S. 918 - 924. http://www.jstor.org/stable/2006496. – ISSN 00255718, 10886842

[33] Bernstein, Daniel J.: *How do I validate Curve25519 public keys?* https://cr.yp.to/ecdh.html#validate, . – [Online; zuletzt aufgerufen am 20.09.2021]

[34] Aumasson, Jean-Philippe: *Should Curve25519 keys be validated?* https://research.kudelskisecurity.com/2017/04/25/should-ecdh-keys-be-validated/, Apr 2017. – [Online; zuletzt aufgerufen am 20.09.2021]

[35] Duong, Thai: *Why not validate Curve25519 public keys could be harmful.* https://vnhacker.blogspot.com/2015/09/why-not-validating-curve25519-public.html, Sep 2015. – [Online; zuletzt aufgerufen am 20.09.2021]

[36] Barker, Elaine ; Chen, Lily ; Roginsky, Allen ; Vassilev, Apostol ; Davis, Richard: *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography.* https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar2.pdf, Mai 2013

[37] Jager, Tibor ; Schwenk, Jörg ; Somorovsky, Juraj: Practical Invalid Curve Attacks on TLS-ECDH. In: *Proceedings, Part I, of the 20th European Symposium on Computer Security – ESORICS 2015 - Volume 9326.* Berlin, Heidelberg : Springer-Verlag, 2015. – ISBN 9783319241739, S. 407–425

[38] Proos, John ; Zalka, Christof: Shor's Discrete Logarithm Quantum Algorithm for Elliptic Curves. In: *Quantum Information & Computation* 3 (Feb 2003). http://dx.doi.org/10.26421/QIC3.4-3. – DOI 10.26421/QIC3.4–3

[39] Galbraith, Steven D. ; Petit, Christophe ; Shani, Barak ; Ti, Yan B.: *On the Security of Supersingular Isogeny Cryptosystems.* Cryptology ePrint Archive, Report 2016/859, 2016. – https://ia.cr/2016/859

[40] Biasse, Jean-Francois ; Jao, David ; Sankar, Anirudh: *A quantum algorithm for computing isogenies between supersingular elliptic curves.* https://cacr.uwaterloo.ca/techreports/2014/cacr2014-24.pdf, Okt 2014

[41] Galbraith, Steven: *Constructing Isogenies Between Elliptic Curves Over Finite Fields.* LMS Journal of Computation and Mathematics, Okt 2000

[42] Kuperberg, Greg: *Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem.* 8th Conference on the Theory of Quantum Computation, Communication and Cryptography 22, Dez 2011. – https://arxiv.org/pdf/1112.3333.pdf

[43] Petit, Christophe: *Faster Algorithms for Isogeny Problems using Torsion Point Images.* Cryptology ePrint Archive, Report 2017/571, Jun 2017. – https://ia.cr/2017/571

[44] Meyer, Michael ; Reith, Steffen: *A faster way to the CSIDH.* Cryptology ePrint Archive, Report 2018/782, Aug 2018. – https://ia.cr/2018/782

[45] Bonnetain, Xavier ; Schrottenloher, André: *Quantum Security Analysis of CSIDH.* Cryptology ePrint Archive, Report 2018/537, Jun 2018. – https://ia.cr/2018/537

[46] Bernstein, Daniel J. ; Lange, Tanja ; Lorenz, Chloe M. ; Panny: *Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies.* Cryptology ePrint Archive, Report 2018/1059, Okt 2018. – https://ia.cr/2018/1059

Abbildungsverzeichnis

2.1 Punktaddition auf elliptischen Kurven 8
2.2 Punktverdopplung auf elliptischen Kurven 9

3.1 Vergleich der Post-Quantum-Verfahren 17

4.1 Diffie-Hellman Schlüsselaustausch 19
4.2 Elliptic Curve Diffie-Hellman 20
4.3 Protokoll eines Supersingular Isogeny Diffie-Hellmann 22

5.1 Diffie-Hellman Datenreihe 27
5.2 Curve25519 Datenreihe 28
5.3 SIDH Datenreihe 29
5.4 CSIDH Datenreihe 30
5.5 Mittelwerte der benötigten CPU-Zyklen während der Testläufe 30

6.1 Beziehung der Risikobewertungen zur Sicherheit des Verfahrens . . . 37
6.2 Einordnung von Pohlig-Hellmann nach DREAD 38
6.3 Einordnung eines fehlerhaft gewählten Generatorelements nach DREAD 39
6.4 Einordnung der Vorausberechenbarkeit nach DREAD 40
6.5 Einordnung von Man-in-the-Middle nach DREAD 41
6.6 Einordnung von Shor's Algorithmus nach DREAD 42
6.7 Mittelwert der Angriffsvektoren auf DH nach DREAD 42
6.8 Einordnung der Pollard-Rho-Methode nach DREAD 44
6.9 Einordnung der ECDH-Schlüsselvalidierung nach DREAD 45
6.10 Einordnung von Man-in-the-Middle nach DREAD 46
6.11 Einordnung von Shor's Algorithmus nach DREAD 47
6.12 Mittelwert der Angriffsvektoren auf ECDH nach DREAD 47
6.13 Einordnung einer fehlerhaft gewählten SIDH-Primzahl nach DREAD 49
6.14 Einordnung der Wiederverwendung von SIDH-Schlüsseln nach DREAD 50
6.15 Einordnung des Meet-in-the-Middle-Verfahrens nach DREAD 51
6.16 Mittelwert der Angriffsvektoren auf SIDH nach DREAD 52
6.17 Einordnung des Hidden-shift-Problems nach DREAD 53
6.18 Einordnung des Meet-in-the-Middle-Verfahrens nach DREAD 54
6.19 Mittelwert der Angriffsvektoren auf CSIDH nach DREAD 54
6.20 Vergleich der Risikowerte der verschiedenen Verfahren nach DREAD . 55

Abkürzungen

AES	Advanced Encryption Standard
CSIDH	Commutative Supersingular Isogeny Diffie Hellman
DES	Data Encryption Standard
DH	Diffie Hellman
DLP	Discrete Logarithm Problem
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie Hellman
ECDLP	Elliptic Curve Discrete Logarithm Problem
IETF	Internet Engineering Task Force
NIST	National Institute of Standards and Technology
NSA	National Security Agency
PQC	Post Quantum Cryptography
SHA	Secure Hash Algorithm
SIDH	Supersingular Isogeny Diffie Hellman
SSH	Secure Shell
TLS	Transport Layer Security

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die hier vorliegende Arbeit selbstständig, ohne unerlaubte fremde Hilfe und nur unter Verwendung der aufgeführten Hilfsmittel angefertigt habe.

Ort, Datum Unterschrift