h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Hochschule Darmstadt

– Fachbereich Informatik –

Performanz Evaluation von PQC in der Authentifizierungsphase von TLS 1.3 unter variierenden Netzwerkcharakteristiken

Abschlussarbeit zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

vorgelegt von

Ronja Wolf

Matrikelnummer: 761118

Referent	:	Prof. Dr. Andreas Heinemann
Korreferent	:	Johanna Henrich

ERKLÄRUNG

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 12. Dezember 2022

Ronja Wolf

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

AES Advanced Encryption Standard

CRYSTALS Cryptographic Suite for Algebraic Lattices

CBC Cipher Block Chaining

DES Data Encryption Standard

ECDH Elliptic-curve Diffie-Hellman

ECDSA Elliptic Curve Digital Signature Algorithm

EdDSA Edwards-curve Digital Signature Algorithm

FALCON Fast Fourier lattice-based compact signatures over NTRU

GAN Global Area Network

IETF Internet Engineering Task Force

MAC Message Authentication Code

NIST National Institute of Standards and Technology

OQS Open Quantum Safe

OSI Open Systems Interconnection

PAN Personal Area Network

PQC Post-Quanten-Kryptographie

RSA Rivet-Shamir-Adleman

SHA Secure Hash Algorithm

TLS Transport Layer Security

Teil I

THESIS

1 EINLEITUNG

In Transport Layer Security (TLS) 1.3 wird ein Handshake durchgeführt, bei dem ein Schlüsselaustausch sowie die Authentifizierung des Servers gegenüber dem Client durchgeführt wird. Bei Bedarf kann auch eine beidseitige Authentifizierung durchgeführt werden. Erst im Anschluss findet die eigentliche Kommunikation statt. Aktuell werden dabei für die Authentifizierung die klassischen asymmetrischen Verfahren Edwards-curve Digital Signature Algorithm (EdDSA), Elliptic Curve Digital Signature Algorithm (ECDSA) und Rivet-Shamir-Adleman (RSA) verwendet. In dieser Arbeit soll betrachtet werden, inwiefern sich die Performanz von TLS 1.3 verändert, wenn stattdessen Post-Quanten-Kryptographie (PQC) Verfahren, d. h. kryptographische Verfahren, die gegen Quantencomputer resistent sind, eingesetzt werden.

1.1 MOTIVATION

Die Entwicklung von Quantencomputern schreitet voran. Wenn Quantencomputer jedoch einsatzfähig sind, wird es mit ihnen möglich sein, viele der aktuell verwendeten kryptographischen Algorithmen zu brechen. Dies liegt daran, dass diese auf Berechnungen basieren, die auf klassischen Computern nicht in einer vertretbaren Zeit durchführbar sind. Dies sind vor allem die Primfaktorzerlegung und der diskrete Logarithmus. Quantencomputer können diese Berechnungen aufgrund ihrer Eigenschaften jedoch schneller durchführen. Davon sind auch die im TLS 1.3 Handshake verwendeten Algorithmen betroffen.

Um zu verhindern, dass hierdurch Probleme entstehen, müssen rechtzeitig PQC-Algorithmen identifiziert werden, mit denen die dann unsicherern Algorithmen ersetzt werden können. Dabei ist zu beachten, dass diese andere Charakteristika haben. Damit der Einsatz von PQC-Algorithmen gelingen kann, muss bekannt sein, wie sich diese unter verschiedenen Netzwerkbedingungen verhalten.

1.2 PROBLEMSTELLUNG UND ZIEL

In dieser Arbeit sollen die klassischen kryptographischen Algorithmen, die bei TLS 1.3 zur Authentifizierung verwendet werden, durch PQC-Algorithmen, wie z. B. Fast Fourier lattice-based compact signatures over NTRU (FALCON), ersetzt werden. In dem Auswahlprozess der National Institute of Standards and Technology (NIST) haben sich dabei besonders Cryptographic Suite for Algebraic Lattices (CRYSTALS) Dilithium, FALCON und SPHINCS+ als vielversprechende Kandidaten für Algorithmen zur digitalen Signatur hervorge-

tan.

Allerdings ist in der realen Welt kein ideales Netzwerk vorhanden und es treten Störfaktoren, wie etwa der Verlust von Paketen oder Latenz auf. Diese Störfaktoren haben einen Einfluss auf die Performanz des Algorithmus. Da TLS 1.3 jedoch ein Protokoll ist, das an verschiedenen Stellen in der Praxis zum Einsatz kommt und hier die Performanz durchaus eine wichtige Rolle spielt, ist die Performanz für den Benutzer von großer Bedeutung.

In dieser Arbeit soll zunächst nur die Authentifizierung mittels PQC-Verfahren durchgeführt werden. Es wird angenommen, dass für den Schlüsselaustausch ein klassisches Verfahren verwendet wird. Dieses bleibt bei allen Versuchen unverändert, sodass nur Auswirkungen durch die Verwendung von PQC-Algorithmen in der Authentifizierungsphase auf die Performanz des TLS-Handshake bewertet werden.

Die konkrete Fragestellung der Arbeit ist, inwiefern sich die Performanz des Handshakes von TLS 1.3 unter der Verwendung verschiedener PQC-Algorithmen in der Authentifizierungsphase verändert, wenn verschiedene Netzwerkparameter manipuliert werden.

Ziel der Arbeit ist es, daraus ableiten zu können, welche Algorithmen für die Verwendung in der Authentifizierungsphase von TLS 1.3 unter realen Netzwerkbedingungen geeignet sind. Dies ist relevant für die Verwendung des Algorithmus in einer Netzwerkumgebung.

1.3 RELATED WORK

Crockett et al. entwarfen 2019 eine Implementierung von PQC im TLS 1.3 Handshake. Dabei wurden verschiedene, in der Literatur vorhandene Grundkonzepte diskutiert [CPS19]. Dabei geben sie eine Empfehlung, wie eine Implementierung aussehen kann. In der Arbeit selbst wird darauf hingewiesen, dass eine Prüfung der Algorithmen sowohl unter Veränderung einzelner Netzwerkparameter als auch unter realistischen Netzwerkbedingungen notwendig ist.

Paul et al. untersuchten 2022, wie Zertifkatsketten für die Authentifizierung mit PQC-Verfahren aussehen könnten [Pau+22]. Dabei verwenden sie "Mixed Certificate Chains", bei denen innerhalb derselbsen Zertifikatskette verschiedene Algorithmen angewandt werden. Diese Arbeit soll diese Konzepte aufgreifen, sie betrachtet jedoch, wie sich die Performanz des Handshakes bei der Veränderung einzelner Netzwerkparameter verhält.

Henrich hat in ihrer Masterarbeit untersucht, wie sich die Verwendung von PQC-Algorithmen im Schlüsselaustausch von TLS1.3 unter veschiedenen Netzwerkbedingungen verhält [Hen22]. Sie kommt dabei zu dem Ergeb-

nis, dass insbesondere Kyber, Saber, NTRU und NTRU Prime auch bei variierenden Netzwerkparametern eine sehr gute Performanz haben und teils noch schneller als Elliptic-curve Diffie-Hellman (ECDH) sind. In Analogie dazu soll in dieser Arbeit die Authentifizierung betrachtet werden.

Das dabei verwendete Framework zur Emulation verschiedener Netzwerkszenarien wurde in einer Arbeit von Paquin et al. aus dem Jahr 2020 [PST20] vorgestellt.

Eine weitere Arbeit in diesem Bereich stammt von Sikeridis et al. aus dem Jahr 2020 und beschäftigt sich bereits mit der Performanz von PQC-Verfahren zur Authentikation in TLS 1.3 [SKD20]. In dieser Arbeit wurde der TLS 1.3 Handshake zwischen einem Client und realen Servern in verschiedenen Staaten durchgeführt, um zu überprüfen, inwiefern dies Auswirkungen auf die Performanz der Algorithmen hat. Diese Arbeit soll sich davon abgrenzen, indem sie gezielt einzelne Netzwerkparameter in einer Emulation manipuliert. Damit soll herausgefunden werden, welche Parameter einen entscheidenden Einfluss auf die Performanz haben.

1.4 VORGEHENSWEISE

Für diese Arbeit soll in einem Linux-Kernel der Aufbau einer TLS 1.3 Verbindung zwischen einem Server und einem Client durchgeführt werden. Dies geschieht unter verschiedenen Netzwerkparametern sowie mit verschiedenen PQC-Algorithmen in der Authentifizierungsphase von TLS 1.3.

Dafür wird für verschiedene PQC-Algorithmen eine Reihe an Versuchen durchgeführt, in denen verschiedene Netzwerkcharakteristiken verändert werden. Nach der erfolgreichen Authentifizierung des Servers gegenüber dem Client wird die Verbindung abgebrochen. Die Zeit zwischen dem Aufbau und dem Abbau der Verbindung wird gemessen und dient als Vergleichsgröße. Diese Versuche sollen ebenfalls unter verschiedenen Sicherheitsleveln durchgeführt werden.

Für den Verbindungsaufbau zwischen Client und Server werden Linux-Kernel-Tools verwendet. Die verschiedenen Netzwerkparameter werden mithilfe des Tools NetEm emuliert. Dabei sollen Veränderungen an folgenden Parametern betrachtet werden:

- Duplikate
- Jitter
- Korrupte Pakete
- Latenz

- Paketverlust
- Reordering
- Übertragungsrate

Diese werden zunächst einzeln betrachtet. Da sie in der Realität jedoch gemeinsam auftreten, können in einem späteren Versuch auch verschiedene Netzwerkparameter miteinander kombiniert werden.

Die eigentliche Authentifizierung wird mithilfe der Open Quantum Safe (OQS) Library [SM17] durchgeführt. Aufgrund des aktuellen Stands der NIST-Empfehlungen werden die folgenden Algorithmen betrachtet:

- SPHINCS$^+$
- Dilithium
- Falcon

Gegebenenfalls sollen die Ergebnisse in einer realen Netzwerkumgebung mit einem Server aus einer Cloudinstanz bestätigt werden. Dies soll jedoch nicht Kern der Arbeit sein.

Nach der Durchführung der Versuche sollten ausführliche Daten dazu vorliegen, wie sich die Laufzeit von TLS 1.3 unter verschiedenen Algorithmen, Algorithmenparametern und Netzwerkcharakteristiken verhält. Diese sollen im Anschluss miteinander verglichen werden.

Dabei werden sich voraussichtlich große Unterschiede zwischen den betrachteten Algorithmen zeigen. Aus dem Vergleich der Performanz von TLS 1.3 soll eine Empfehlung gegeben werden, welche Algorithmen für welche Art von Netzwerk am besten geeignet sind.

2 GRUNDLAGEN

2.1 NETZWERKE

Ein Computernetzwerk besteht mindestens aus zwei Computern, einem Übertragungsmedium und Protokollen, die festlegen, in welcher Form die Computer Daten austauschen Ein Computernetz kann sich physisch über einige Meter (dann spricht man von einem Personal Area Network (PAN)) bis über die gesamte Erde (dann spricht man von einem Global Area Network (GAN)) ausdehnen. Das Internet ist ein GAN. [Bau18a].

Beim Open Systems Interconnection (OSI) Referenzmodell (vgl. Abbildung 2.1) wird eine Netzwerkverbindung in sieben logische Schichten unterteilt. Es wird verwendet, um eine Kommunikation über ein Netzwerk darzustellen. Die einzelnen Schichten sind dabei durch Schnittstellen miteinander verbunden, sodass einzelne Protokolle leicht ausgetauscht werden können [Bau18b]

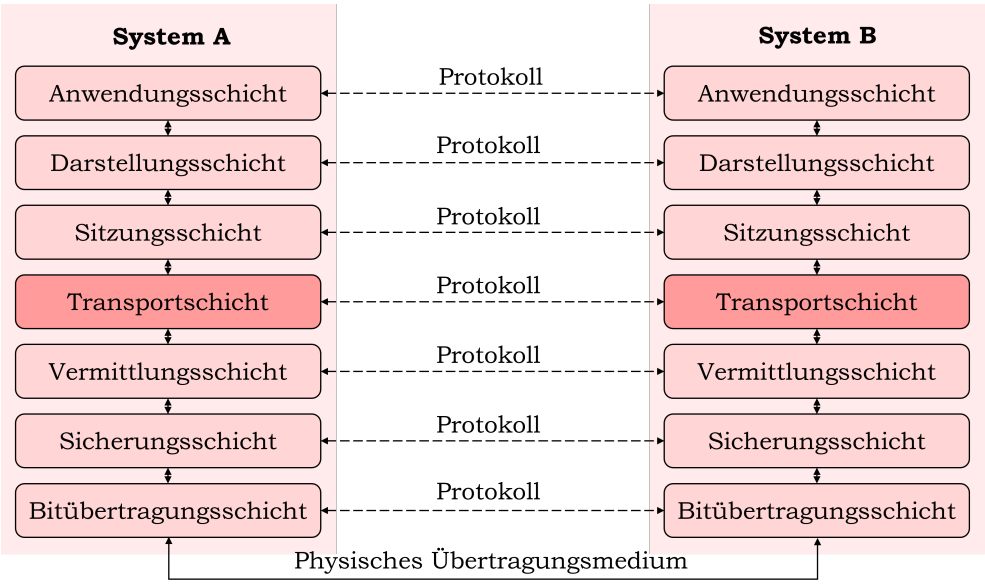

Abbildung 2.1: OSI-Referenzmodell mit hervorgehobener Transportschicht

2.2 KRYPTOGRAPHIE

Der Begriff Kryptographie bezeichnet die "Wissenschaft vom geheimen Schreiben" [Wät18]. Dabei wird der verschlüsselte Text als "Chiffretext" und der entschlüsselte Text als "Klartext" bezeichnet. Das Gegenstück der Kryptogra-

phie ist die Kryptanalyse, deren Ziel es ist, Kryptographie zu brechen. Kryptographie und Kryptanalyse bilden das Fachgebiet der Kryptologie [PP10b].

Ein wichtiges Prinzip der Kryptographie ist das Kerckhoffsche Prinzip. Dieses besagt, dass die verwendeten Methoden zur Ver- und Entschlüsselung auch dann noch sicher sein müssen, wenn sie öffentlich bekannt sind. Dies bedeutet, dass die Sicherheit des Verfahrens auf dem Schlüssel basieren muss.

Dies ist aus mehreren Gründen entscheidend. Zum einen ist es i. d. R. nicht oder nur mit großen Schwierigkeiten möglich, ein gesamtes Verfahren unter Verschluss zu halten. Die Geheimhaltung eines Schlüssels ist deutlich einfacher. Zum anderen ist es so möglich, das Verfahren durch Dritte evaluieren zu lassen und auf Fehler hingewiesen werden zu können [KW11a].

Innerhalb der Kryptographie unterscheidet man dann zwischen symmetrischer und asymmetrischer Kryptographie [PP10b]. Diese werden im Folgenden näher betrachtet.

2.2.1 *Symmetrische Kryptographie*

Bis 1976 wurde ausschließlich mit symmetrischer Kryptographie gearbeitet. Klassische Einsatzgebiete dieser Art der Verschlüsselung sind auch heute noch die Datenverschlüsselung sowie Integritätsprüfungen.

Bei der symmetrischen Kryptographie werden für die Ver- und Entschlüsselung der gleiche Schlüssel verwendet. Der zu verschlüsselnde Klartext wird mit dem Schlüssel verschlüsselt und kann dann über einen unsicheren Kanal transportiert werden. Der Empfänger erhält den Chiffretext und kann diesen mithilfe des gleichen Schlüssels wieder in den Klartext umwandeln.

Ein großes Problem dieser Art der Verschlüsselung ist das Problem des sicheren Schlüsselaustauschs. Die beiden Kommunikationspartner müssen sich auf einen Schlüssel einigen. Dies muss jedoch über einen sicheren Kanal geschehen, damit sichergestellt werden kann, dass der Schlüssel nicht an unberechtigte Dritte gerät [PP10b].

Innerhalb der symmetrischen Kryptographie kommen Strom- und Blockchiffren zum Einsatz. Während Stromchiffren jedes Bit des Klartexts einzeln verschlüsseln, verschlüsseln Blockchiffren den Klartext in Blöcken fester Größe [PP10e].

Stromchiffren sind besonders für Echtzeitanwendungen und für den Einsatz auf Systemen mit begrenzten Ressourcen geeignet. Dies liegt daran, dass sie hereinkommende Bits sofort verschlüsseln und über einen unsicheren Kanal übertragen können. Sie sind nicht darauf angewiesen, eine be-

stimmte Menge an Daten abzuwarten. Eine bekannte Stromchiffren ist beispielsweise RC4, die bei HTTPS zum Einsatz kommt.

In den meisten Anwendungen, in denen symmetrische Kryprographie zum Einsatz kommt, werden jedoch Blockchiffren verwendet [PP10e]. Auf Blockchiffren bauen viele weitere Bausteine der Kryptographie auf. So können beispielsweise Stromchiffren oder Hashfunktionen durch Blockchiffren erzeugt werden.

2.2.2 *Asymmetrische Verschlüsselung*

Im Gegensatz zur symmetrischen Kryptographie wird bei der asymmetrischen Kryptographie nicht derselbe Schlüssel für Ver- und Entschlüsselung verwendet. Der Grundgedanke ist, dass der Schlüssel für die Entschlüsselung geheim gehalten werden muss, der Schlüssel für die Verschlüsselung jedoch nicht.

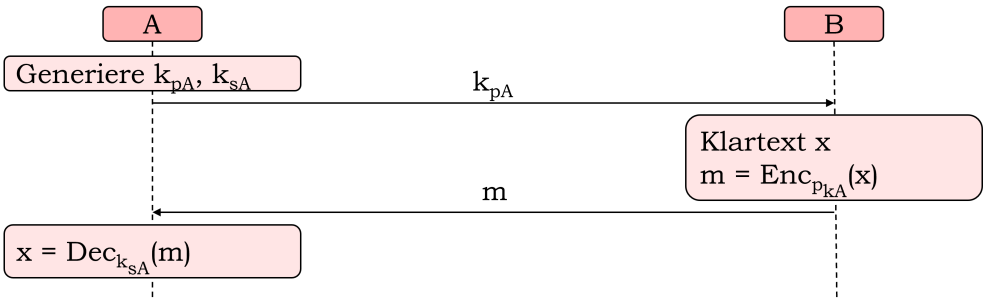

Abbildung 2.2: Ein einfaches asymmetrisches kryptographisches Verfahren

Asymmetrische kryptographische Verfahren basieren jeweils auf einem von drei mathematischen Problemene:

1. Primfaktorzerlegung
2. Diskreter Logarithmus
3. Elliptische Kurven

Alle diese Probleme haben gemein, dass sie ohne Zusatzinformationen nur mit einem sehr großen Zeitaufwand durch einen Computer gelöst werden können. Die Zusatzinformation, die das Lösen des Problems und somit das Entschlüsseln der Nachricht ermöglicht, ist der private Schlüssel des Empfängers der Nachricht [PP10c].

Ein bekanntes Beispiel für eine asymmetrische Chiffre, die auf dem mathematischen Problem der Primfaktorzerlegung basiert, ist RSA (vgl. Abschnitt 2.2.4.3.

Der diskrete Logarithmus ist beispielsweise Grundlage des Diffie-Hellman-Schlüsselaustauschs 2.3. Für diesen gibt es auch eine Variante, die elliptische

Kurven verwendet.

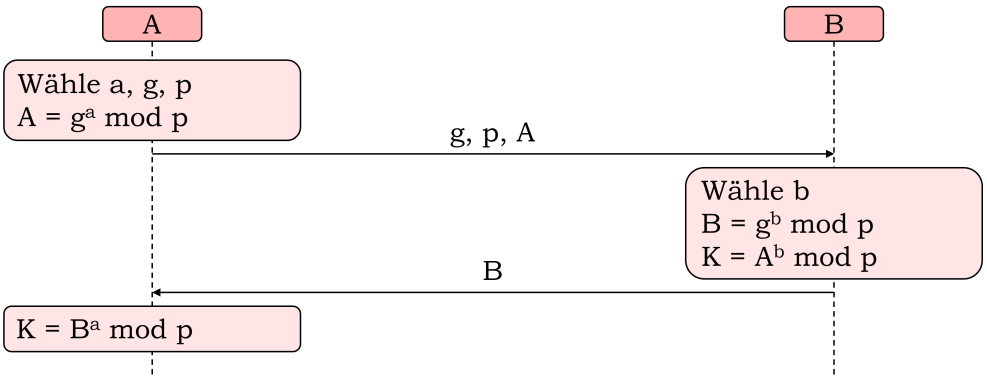

Abbildung 2.3: Funktionsweise des Diggie-Hellman-Schlüsselaustauschs

2.2.3 *Kryptographische Hashfunktionen*

Hashfunktionen sind kryptographische Funktionen, die eine Eingabe beliebiger Länge auf eine Ausgabe fester Länge abbildet. Diese Berechnung muss sehr einfach erfolgen können. Auf der Ausgabe können jedoch keine Schlüssel auf die Eingabe geschlossen werden.

Weiterhin darf zu einem bekannten Eingabewert keine zweite Eingabe gefunden werden können, die den gleichen Hashwert ergibt. Diese Eigenschaft bezeichnet man als schwache Kollisionsresistenz.

Außerdem gibt es die starke Kollisionsresistenz. Diese besagt, dass keine frei wählbare unterschiedlichen Eingaben gefunden werden dürfen, die den gleichen Hashwert ergeben.

Hashfunktionen können entweder eine eigene designierte Hashfunktion sein oder auf einer symmetrischen Blockchiffre basieren. Designierte Hashfunktionen sind beispielsweise Secure Hash Algorithm (SHA)-2 und SHA-3.

2.2.4 *Authentifizierungsverfahren*

Auch bei den Authentifizierungsverfahren unterscheidet man zwischen symmetrischen Verfahren und asymmetrischen Verfahren. Auch diese sollen im folgenden genauer betrachtet werden.

2.2.4.1 *Symmetrische Authentifizierungsverfahren*

Symmetrische Authentifizierungsverfahren werden auch als Message Authentication Code (MAC) bezeichnet. Sie basieren in der Regel auf Blockchiffren (vgl. Abschnitt 2.2.1) oder kryptographischen Hashfunktionen (vgl.

Abschnitt 2.2.3) [KW11b].

MACs generieren aus einer Nachricht beliebiger Länge eine kryptographische Prüfsumme, die unabhängig von der Länge der Nachricht immer die gleiche Länge hat. Da es sich hier um ein symmetrisches Verfahren handelt, benötigen die Kommunikationspartner einen gemeinsamen kryptographischen Schlüssel.

Die Prüfsumme erlaubt es, Manipulationen an der Nachricht zu erkennen sowie den Ursprung der Nachricht nachzuvollziehen. Mit MACs ist es jedoch nicht möglich, die Urheberschaft einer Nachricht eindeutig einer Person zuzuordnen. Es kann lediglich festgestellt werden, dass die Person den Schlüssel kennt. Diesen kennen jedoch immer mindestens zwei Parteien.

MACs können auf Blockchiffren basieren. Häufig wird beispielsweise AES im Cipher Block Chaining (CBC) Modus dafür verwendet. Dafür werden für die einzelnen Blöcke der zu authentifizierenden Nachricht die folgenden Berechnungen durchgeführt:

$$y_1 = Enc_k(x_1 \oplus IV)$$

$$y_2 = Enc_k(x_2 \oplus y_1)$$

$$y_i = Enc_k(x_i \oplus y_{i-1})$$

Ein einfaches Verfahren für einen MAC, der auf einer Hashfunktion basiert, ist der HMAC. In der Abbildung wird der Schlüssel als geheimer Präfix verwendet. Dieser könnte jedoch auch als Suffix verwendet werden. [PP10d]

2.2.4.2 *Asymmetrische Authentifizierungsverfahren*

Asymmetrische Authentifizierungsverfahren werden auch als Digitale Signaturen bezeichnet. Dabei verschlüsselt die zu authentifizierende Partei einen Klartext mit ihrem privaten Schlüssel und schickt den Klartext und den Chiffretext an den Kommunikationspartner. Wenn dieser den Chiffretext mit dem öffentlichen Schlüssel der zu authentifizierenden Person entschlüsselt und den korrekten Klartext erhält, ist die Authentifizierung erfolgreich verlaufen.

Das erste praktisch einsetzbare Verfahren zur digitalen Signatur wurde von Ronald Rivest, Adi Shamir und Len Adleman in einem Paper beschrieben. Dieses war jedoch noch nicht wirklich sicher, sondern zeigte lediglich, wie eine digitale Signatur prinzipiell umgesetzt werden könnte.

Ein großes Problem von digitalen Signaturen ist, dass sichergestellt werden muss, dass der öffentliche Schlüssel auch wirklich zum Kommunikati-

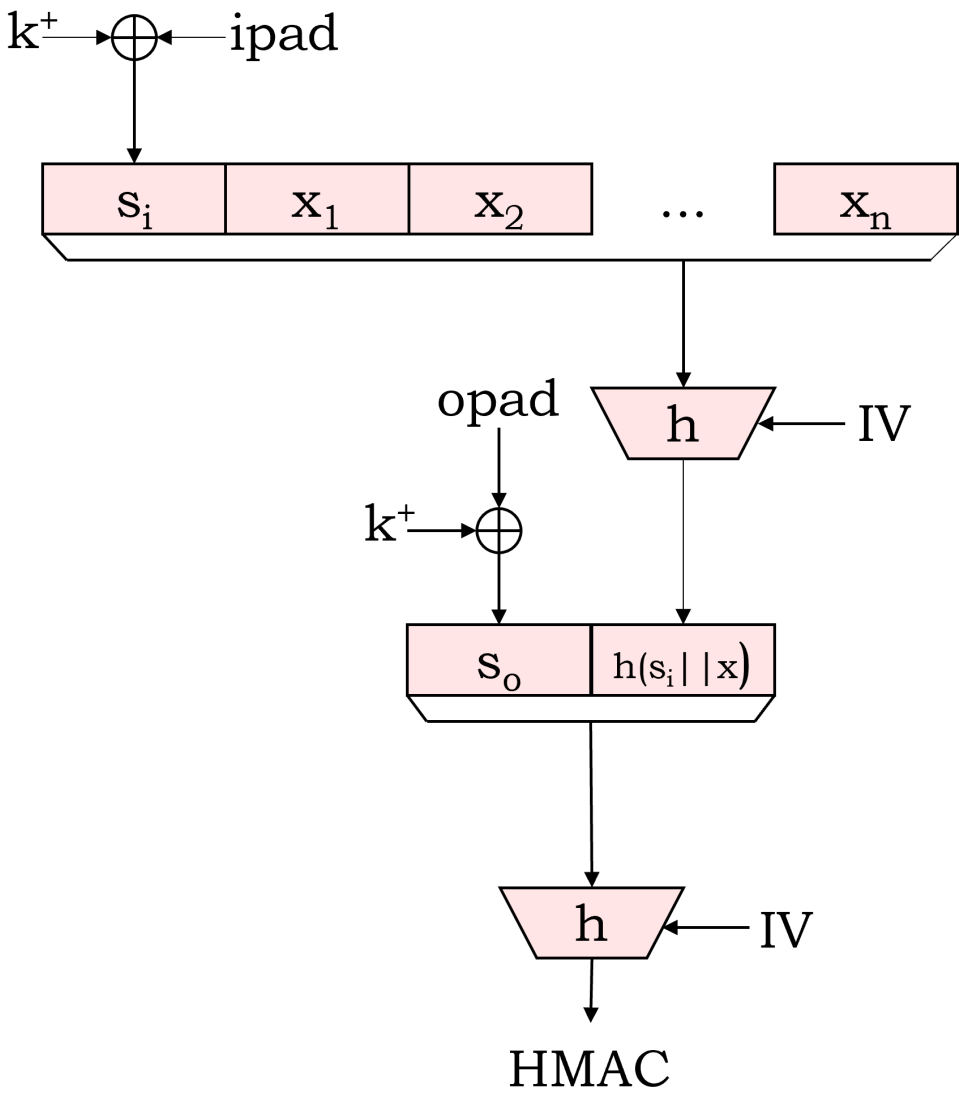

Abbildung 2.4: Ein einfaches HMAC-Verfahren

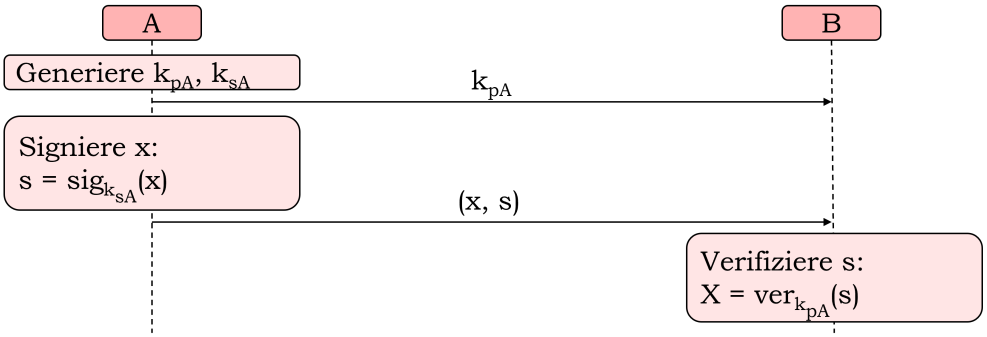

Abbildung 2.5: Ein einfaches Verfahren zur digitalen Signatur

onspartner gehört. Dafür werden Zertifikate verwendet (vgl. Abschnitt 2.2.5) [PP10a].

In den folgenden Abschnitten 2.2.4.3 bis **??** werden einige Authentifizierungsverfahren kurz vorgestellt.

2.2.4.3 *RSA*

RSA wird heute vor allem für das Verschlüsseln kleinerer Datenmengen (beispielsweise für den Transport kryptogrpahischer Schlüssel) und digitale Signaturen eingesetzt. Die Verschlüsselung mit RSA basiert auf dem Problem der Primfaktorzerlegung. Dies basiert darauf, dass es sehr effizient möglich ist, große Zahlen miteinander zu multiplizieren. Ein einfaches Faktorisierungsverfahren ist jedoch nicht bekannt.

Die Funktionsweise von RSA wird in Abbildung 2.6 gezeigt.

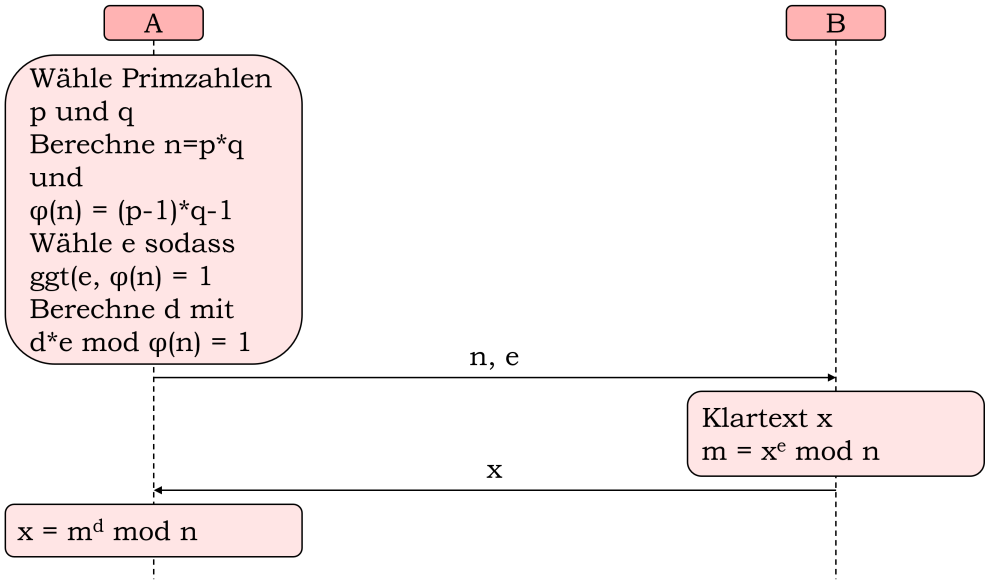

Abbildung 2.6: Funktionsweise von RSA

RSA kann genauso zur digitalen Signatur verwendet werden. Dafür muss die gewünscht Nachricht mit dem privaten Schlüssel d signiert werden. Der Kommunikationspartner kann die Signatur dann mit dem öffentlichen Schlüssel e verifizieren.

Dies ist aufgrund der Potenzregeln möglich:

$$(x^e)^d = x^{ed} = x^{de} = (x^d)^e \equiv h(\mod n)$$

Die Sicherheit von RSA hängt dabei stark von der Länge der gewählten Primzahlen ab. 2020 war die größte öffentlich bekannte RSA-Zahl n, die faktorisiert werden konnte, 829 Bit lang. In der Praxis sind RSA-Schlüssel in

der Regel zwischen 1024 und 4096 Bit lang. Aktuelle Empfehlungen sprechen sich für eine Schlüssellänge von mindestens 2048 Bit aus.

RSA ist nur so lange sicher, wie es keinen effizienten Algorithmus zur Primzahlzerlegung gibt. Mit dem 1994 von Peter Shor vorgestellten Algorithmus ist dies, zumindest wenn man den Einsatz von Quantencomputern berücksichtigt, nicht mehr gegeben. Shors Algorithmus kann das Problem in Polynomialzeit lösen (vgl. Kapitel 2.4.4).

2.2.4.4 *Rabin*

Das Rabin-Kryptosystem ist eng mit RSA verwandt und basiert ebenfalls auf dem Faktorisierungsproblem. Dabei werden zur Schlüsselerzeugung folgende Berechnungen ausgeführt:

$$p \equiv 3 \mod 4$$

$$q \equiv 3 \mod 4$$

$$n = p * q$$

Der öffentliche Schlüssel ist dann n. Der geheime Schlüssel ist (p, q).

Um eine Nachricht zu verschlüsseln, muss diese zunächst auf einen Wert m < n abgebildet werden. Anschließend wird der Chiffretext c berechnet als $c = m^2 \mod n$.

Zur Entschlüsselung kann der chinesische Restsatz herangezogen werden, sofern p und q, die den geheimen Schlüssel bilden, bekannt sind. Dabei erhält man jedoch vier mögliche Klartexte. Welcher dieser vier möglichen Klartexte der richtige ist, muss erraten werden. Dies ist nur möglich, wenn der Klartext einer bestimmten Struktur (etwa einer Sprache) folgt. Ist dies nicht der Fall, muss eine solche Struktur künstlich erzeugt werden. Dies verringert jedoch die Sicherheit der Chiffre. Aus diesem Grund hat Rabin in der Praxis nur wenige Einsatzfelder.

Auch digitale Signaturen können analog zu RSA durchgeführt werden.

Anders als bei RSA kann für das Rabin-Kryptosystem bewiesen werden, dass die Schwierigkeit der Entschlüsselung genauso schwer ist wie das Faktorisierungsproblem selbst. Daher ist Rabin noch ein wenig sicherer als RSA.

2.2.4.5 *Lamport-Einmal-Signaturverfahren*

Ein weiterer Algorithmus, der zur digitalen Signatur verwendet werden kann, ist das Einmal-Signaturverfahren, das 1979 von Leslie Lamport entwickelt wurde. Für dieses wird eine Einmalfunktion - dies kann beispielsweise

eine Hashfunktion sein - benötigt. Im folgenden Beispiel wird eine 256-Bit-Hashfunktion verwendet.

Um den privaten Schlüssel zu generieren, müssen mit einem Zufallszahlengenerator 265 Paare von Zufallszahlen, d. h. insgesamt 512 Zufallszahlen, die jeweils eine Länge von 256 Bit haben, erzeugt werden. Für den zugehörigen öffentlichen Schlüssel werden die Hashwerte der 512 Zufallszahlen berechnet. Diese 512 Hashwerte sowie die verwendete Hashfunktion müssen veröffentlicht werden.

Um nun eine Nachricht zu signieren, muss zunächst der Hashwert der Nachricht errechnet werden. Dieser hat per Definition 256 Bit. Für jede 0 in der gehashten Nachricht verschickt Alice nun die erste Zahl eines Zahlenpaars ihres geheimen Schlüssels, für jede 1 wählt sie die zweite Zahl des Zahlenpaares.

Um die Signatur zu verifizieren, wird erneut der Hashwert der Nachricht errechnet. Außerdem müssen die Hashwerte der 265 Zufallszahlen des öffentlichen Schlüssels errechnet werden. Ist das erste Bit in der gehashten Nachricht eine 0, muss die erste Zahl des gehashten öffentlichen Schlüssels der ersten Zahl des ersten Zahlenpaares des öffentlichen Schlüssels entsprechen. Ist das erst Bit in der gehashten Nachricht eine 1, muss die zweite Zahl des ersten Zahlenpaares übereinstimmen.

Da hier bei jedem Signaturvorgang die Hälfte des privaten Schlüssels veröffentlicht werden muss, kann mit jeden Schlüssel nur eine Signatur angefertigt werden. Danach ist der gesamte Schlüssel, d. h. auch die nicht verwendeten Zufallszahlen, zu löschen.

Die Sicherheit dieses Signaturverfahrens basiert auf der Sicherheit der verwendeten Einwegfunktion. Wird hier eine schwache Funktion gewählt, so ist auch die Signatur nicht sicher. Wird hingegen eine starke Funktion gewählt, ist das Verfahren sehr sicher.

Aktuell wird davon ausgegangen, dass das Lamport-Einmal-Signaturverfahren auch unter dem Einsatz von Quantencomputern noch sicher ist, solange eine große Hashfunktion verwendet wird.

2.2.4.6 *Merkle*

Digitale Signaturen mithilfe eines Merkle-Baums bauen auf dem Lamport-Einmal-Signaturverfahren auf. Sie ermöglichen jedoch eine endliche Anzahl an digitalen Signaturen mit demselben Schlüssel.

Für dieses Verfahren wird ein Merkle-Baum benötigt (vgl. Abbildung 2.7). Im Merkle-Baum ist jeder Knoten ein Hashwert der Verkettung seiner Kinder. Dies bedeutet beispielsweise:

$$a_{1,0} = H(a_{0,0}||a_{0,1})$$

$$a_{2,0} = H(a_{1,0}||a_{1,1})$$

$$a_{3,0} = H(a_{2,0}||a_{2,1})$$

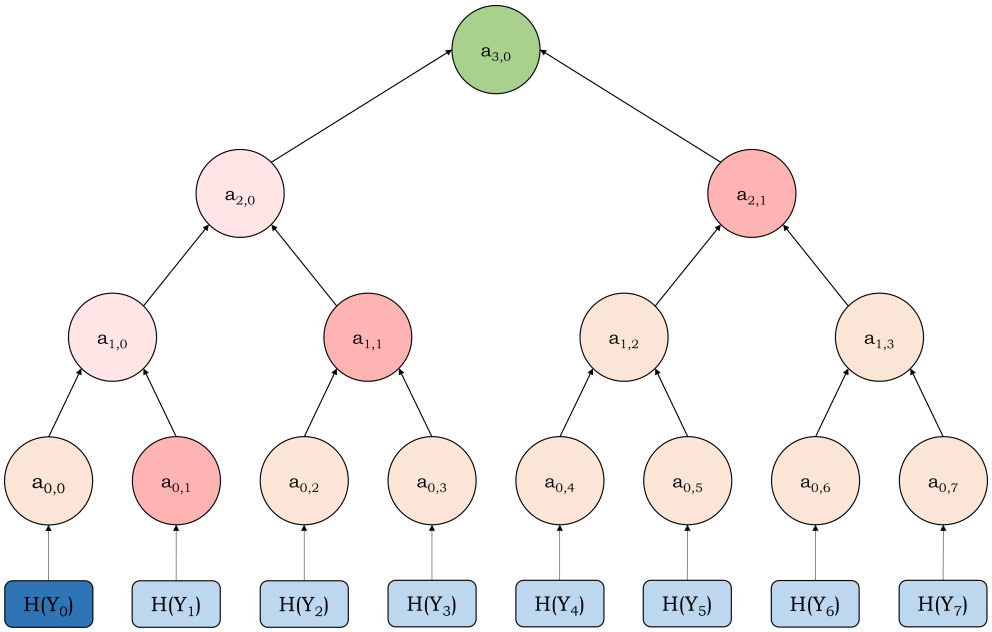

Abbildung 2.7: Digitale Signaturen mithilfe eines Merkle-Baums

In der Abbildung ist beispielhaft ein Merkle-Baum für einen privaten Schlüssel aus acht Zufallszahlen dargestellt. Die einzelnen Zufallszahlen des Schlüssels Y_i werden gehasht und bilden dann die Blätter des Merkle-Baums. Der öffentliche Schlüssel X ist die Wurzel des Baumes.

Um nun eine Signatur prüfen zu können, wird nur eine Zufallszahl des geheimen Schlüssels benötigt. Zusätzlich müssen einige Knoten (in der Abbildung sind dies $a_{0,1}$, $a_{1,1}$ und $a_{2,1}$) zusätzlich übermittelt werden.

2.2.4.7 *GMR*

2.2.4.8 *DSA*

2.2.5 *Kryptographische Zertifikate*

Ein kryptographisches Zertifikat bindet einen öffentlichen Schlüssel an eine Entität. Dies kann eine natürliche Person oder auch eine Organisation oder ein System sein.

Digitale Zertifikate enthalten in der Regel die folgenden Informationen:

1. den Namen des Ausstellers

2. die Rahmenbedingungen unter denen das Zertifikat ausgestelle wurde
3. die Gültigkeitsdauer des Zertifikats
4. den öffentlichen Schlüssel des Eigentümers
5. den Namen des Eigentümers
6. ggf. weitere Informationen zum Eigentümer
7. Zulässige Anwendungs- und Geltungsbereiche des Schlüssels
8. eine digitale Signatur des Ausstellers über die o. g. Informationen

Ein häufig verwendeter Standard für Zertifikate ist X.509. Dieser wird auch in TLS verwendet (vgl. Abschnitt 2.3).

2.3 TRANSPORT LAYER SECURITY

TLS ist ein 1996 durch die Internet Engineering Task Force (IETF) festgelegtes Protokoll mit dem Ziel, einen Standard für die sichere Kommunikation im über das Internet zu finden. Das Protokoll agiert auf der Transportschicht. Während die ursprünglichen Ziele des Protokolls Datensicherheit und -integrität waren, wurde in späteren Versionen zunehmend auch die Kommunikation mit anderen TLS-Versionen sowie die Möglichkeit des Ausbaus des Protokolls, z. B. durch neue Algorithmen, betrachtet [Kiz20].

2.4 POST-QUANTUM-KRYPTOGRAPHIE UND NIST

2.4.1 *Quanteninformatik*

Während die klassische Einheit in der Informatik aus einem Bit entstehen kann, das entweder den Wert 0 oder den Wert 1 annehmen kann, wird die Grundlage der Quanteninformatik durch sogenannte Qubits gebildet. Diese unterscheiden sich in einigen wesentlichen Punkten von den klassischen Bits [Jus20]:

1. Während der Wert eines klassischen Bits zu jedem Zeitpunkt eindeutig als 0 oder 1 identifiziert werden kann, können Qubits auch einen Wert zwischen 0 und 1 annehmen.
2. Wird der Wert eines Qubits gemessen, ist dieser immer 0 oder 1. Der Wert eines Qubits kann sich demnach durch die Messung verändern.
3. Eine Veränderung in einem Qubit kann eine instantane Veränderung in einem anderen Qubit hervorrufen.

2.4.2 *Kryptoagilität*

Ein wichtiges Konzept in diesem Zusammenhang ist das Konzept der Kryptoagilität. Durch die fortschreitende Entwicklung sind immer wieder etablierte kryptographische Verfahren unsicher geworden. Ist dies der Fall, müssen diese ersetzt werden.

2.4.3 *National Institute of Standards and Technology*

Das National Institute of Standards and Technology (übersetzt Nationales Institut für Standards und Technoligue) ist eine Bundesbehörde der Vereinigten Staaten, deren Hauptaufgabe Standardisierungsprozesse sind. In der Kryptographie war das NIST u. a. dafür verantwortlich, die Verschlüsselungsstandards Advanced Encryption Standard (AES) und Data Encryption Standard (DES) zu standardisieren.

2.4.4 *Entwicklung der Post-Quantum-Kryptographie*

2.4.5 *NIST-PQC-Verfahren*

2.4.5.1 *Motivation*

Das Internet basiert darauf, dass verschiedene Computersysteme miteinander interagieren können. Damit dies möglich ist, hat die internationale Gemeinschaft sich auf eine Reihe standardisierter Protokolle geeinigt. Einige dieser Protokolle sind durch die fortschreitende Entwicklung von Quantencomputern akut gefährdet.

Ein Beispiel dafür ist RSA. RSA ist ein Algorithmus, der in vielen Protokollen verwendet wird. Er basiert auf dem Problem der Primfaktorzerlegung, d. h. darauf, dass es zwar leicht ist, große Zahlen zu multiplizieren, jedoch sehr schwer ist, eine große Zahl in ihre Primfaktoren zu zerteilen. Mit Shors Algorithmus, der auf Quantencomputern ausgeführt werden kann, ist dies jedoch effizient möglich [ST16].

Auf der anderen Seite gibt es inzwischen eine große Anzahl an vorgeschlagenen Algorithmen, die diese Probleme lösen sollen. Um nun wieder geeignete Algorithmen zu finden, die sich für eine Standardisierung eignen, muss diese Menge an Algorithmen evaluiert werden.

Dies soll durch das NIST-PQC-Verfahren umgesetzt werden. Ein ähnliches Verfahren wurde bereits von 1997 bis 2001 durchgeführt. Das Ergebnis des damaligen Verfahrens war die Standardisierung von AES [Schoo].

LITERATUR

[Bau18a] Christian Baun. "Grundlagen der Computervernetzung". In: *Computernetze kompakt*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, S. 15–33. ISBN: 978-3-662-57469-0. DOI: 10.1007/978-3-662-57469-0_3. URL: https://doi.org/10.1007/978-3-662-57469-0_3.

[Bau18b] Christian Baun. "Protokolle und Protokollschichten". In: *Computernetze kompakt*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, S. 35–44. ISBN: 978-3-662-57469-0. DOI: 10.1007/978-3-662-57469-0_4. URL: https://doi.org/10.1007/978-3-662-57469-0_4.

[CPS19] Eric Crockett, Christian Paquim und Douglas Stebila. *Prototyping post-quantum and hybrid key exchange and authentication in TLS and SSH*. 2019.

[Hen22] Johanna Henrich. "Performanz Evaluation von PQC in TLS 1.3 unter variierenden Netzwercharakteristiken". Magisterarb. Hochschule Darmstadt, 2022.

[Jus20] Bettina Just. "Einführung". In: *Quantencomputing kompakt: Spukhafte Fernwirkung und Teleportation endlich verständlich*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2020, S. 1–4. ISBN: 978-3-662-61889-9. DOI: 10.1007/978-3-662-61889-9_1. URL: https://doi.org/10.1007/978-3-662-61889-9_1.

[Kiz20] Joseph Migga Kizza. "Computer Network Security Protocols". In: *Guide to Computer Network Security*. Cham: Springer International Publishing, 2020, S. 367–398. ISBN: 978-3-030-38141-7. DOI: 10.1007/978-3-030-38141-7_17. URL: https://doi.org/10.1007/978-3-030-38141-7_17.

[KW11a] Ralf Küsters und Thomas Wilke. "Grundlegendes". In: *Moderne Kryptographie: Eine Einführung*. Wiesbaden: Vieweg+Teubner, 2011, S. 7–12. ISBN: 978-3-8348-8288-2. DOI: 10.1007/978-3-8348-8288-2_2. URL: https://doi.org/10.1007/978-3-8348-8288-2_2.

[KW11b] Ralf Küsters und Thomas Wilke. "Symmetrische Authentifizierungsverfahren". In: *Moderne Kryptographie: Eine Einführung*. Wiesbaden: Vieweg+Teubner, 2011, S. 211–237. ISBN: 978-3-8348-8288-2. DOI: 10.1007/978-3-8348-8288-2_9. URL: https://doi.org/10.1007/978-3-8348-8288-2_9.

[PP10a] Christof Paar und Jan Pelzl. "Digital Signatures". In: *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 259–292. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3_10. URL: https://doi.org/10.1007/978-3-642-04101-3_10.

[PP10b] Christof Paar und Jan Pelzl. "Introduction to Cryptography and Data Security". In: *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 1–27. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3_1. URL: https://doi.org/10.1007/978-3-642-04101-3_1.

[PP10c] Christof Paar und Jan Pelzl. "Introduction to Public-Key Cryptography". In: *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 149–171. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3_6. URL: https://doi.org/10.1007/978-3-642-04101-3_6.

[PP10d] Christof Paar und Jan Pelzl. "Message Authentication Codes (MACs)". In: *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 319–330. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3_12. URL: https://doi.org/10.1007/978-3-642-04101-3_12.

[PP10e] Christof Paar und Jan Pelzl. "Stream Ciphers". In: *Understanding Cryptography: A Textbook for Students and Practitioners*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, S. 29–54. ISBN: 978-3-642-04101-3. DOI: 10.1007/978-3-642-04101-3_2. URL: https://doi.org/10.1007/978-3-642-04101-3_2.

[PST20] Christian Paquin, Douglas Stebila und Goutam Tamvada. "Benchmarking Post-quantum Cryptography in TLS". In: *International Conference on Post-Quantum Cryptography*. 2020.

[Pau+22] Sebastian Paul, Norman Lahr, Yulia Kuzovkova und Ruben Niederhagen. "Mixed Certificate Chains for the Transition to Post-Quantum Authentication in TLS 1.3". In: *Proceedings of the 2022 ACM Asia Conference onComputer and Communications Security*. 2022.

[Schoo] Bruce Schneier. "AES Announced". In: *Schneier on Security* (2000). URL: https://www.schneier.com/crypto-gram/archives/2000/1015.html#8.

[SKD20] Dimitrios Sikeridis, Panos Kampanakis und Michael Devetsikiotis. "Post-Quantum Authentication in TLS 1.3: A Performance Study". In: *Network and Distributed Systems Security (NDSS) Symposium 2020*. 2020.

[ST16] National Institute for Standards und Technlology. *Report on Post-Quantum Cryptography*. NISTIR 8105. Apr. 2016. URL: https://csrc.nist.gov/publications/detail/nistir/8105/final#pubs-documentation.

[SM17] Douglas Stebila und Michele Mosca. "Post-quantum key exchange for the Internet and the Open Quantum Safe project". In: *Selected Areas in Cryptography (SAC)*. 2017. URL: https://openquantumsafe.org/.

[Wät18] Dietmar Wätjen. "Grundlagen". In: *Kryptographie: Grundlagen, Algorithmen, Protokolle*. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 1–13. ISBN: 978-3-658-22474-5. DOI: 10.1007/978-3-658-22474-5_1. URL: https://doi.org/10.1007/978-3-658-22474-5_1.