

TRANSPORTSICHERHEIT

Verteilte Systeme

Prof. Dr. Georg Hinkel
31.05.2024

GLIEDERUNG

Datum	Vorlesung	Übungsblatt	Abgabe
19.04.2024	Einführung	HamsterLib	06.05.2024
26.04.2024	Netzwerkprogrammierung	Theorie	
03.05.2024	World Wide Web	HamsterRPC 1	20.05.2024
10.05.2024	Remote Procedure Calls	Theorie	
17.05.2024	Webservices	HamsterRPC 2	03.06.2024
24.05.2024	Fehlertolerante Systeme	Theorie	
31.05.2024	Transportsicherheit	HamsterREST	17.06.2024
07.06.2024	Architekturen für Verteilte Systeme	Theorie	
14.06.2024	Internet der Dinge	HamsterIoT	01.07.2024
21.06.2024	Namen- und Verzeichnisdienste	Theorie	
28.06.2024	Authentifikation im Web	HamsterAuth	15.07.2024
05.07.2024	Infrastruktur für Verteilte Systeme	Theorie	
12.07.2024	Wrap-Up	HamsterCluster (Bonus)	16.08.2024

Agenda

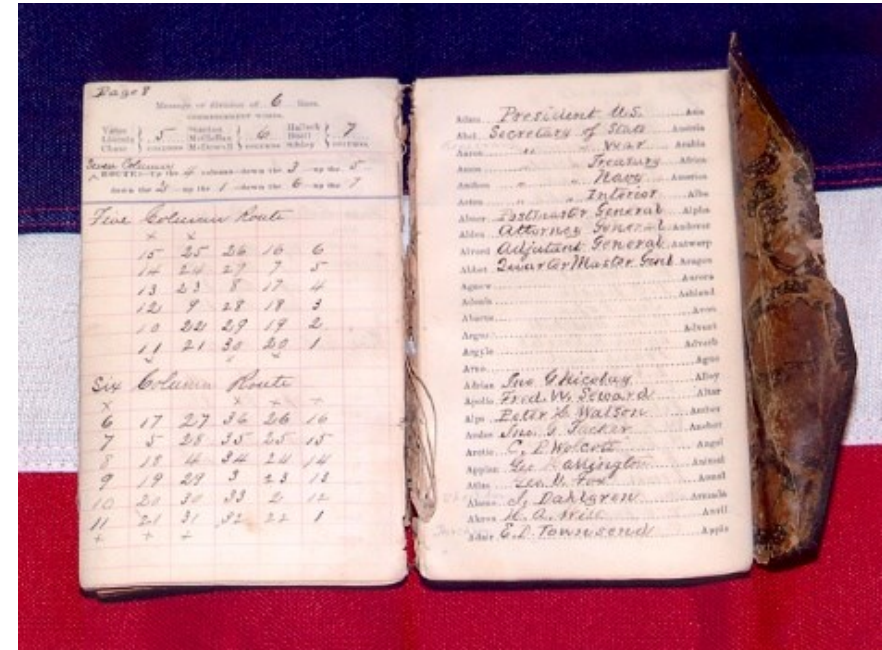
- Verschlüsselung
 - Symmetrisch
 - Asymmetrisch
- Digitale Signaturen
- Schlüsselverwaltung
- Zertifikate
- Transportsicherheit

Lernziele

- Needham-Schröder Protokoll (Kerberos) erklären können
- TLS erklären können, sowohl für TCP als auch für QUIC

VERSCHLÜSSELUNGSVERFAHREN

- Grundproblem: Sichere Nachrichtenübertragung
 - Niemand außer dem Empfänger soll Nachricht lesen können
- Bekannte und sichere Lösung: One-Time-Pad, nutze Schlüssel immer nur ein mal
 - Problem: Schlüsselaustausch
 - Verfahren mit konstantem Schlüssel gesucht
- Viele historische Verfahren (alle gebrochen)
 - Caesar-Code
 - „Le Chiffre indéchiffrable“ (Vigenère)
 - Enigma



Kryptografisches Codebuch aus dem amerikanischen Bürgerkrieg
[<https://de.wikipedia.org/wiki/Verschl%C3%Bcsselung>]

- Symmetrisch
 - Empfänger muss gleichen Schlüssel K besitzen wie der Absender
 - Verschlüsselung $c = E(m, K)$, Entschlüsselung $m = D(c, K)$
- Asymmetrisch
 - Ein Partner muss nur einen Teil des Schlüssels kennen (öffentlicher Schlüssel, K^+)
 - Annahme: Kompletter (privater) Schlüssel K^- ist aus öffentlichem Schlüssel nicht rekonstruierbar
 - Dadurch einfache Verteilung des öffentlichen Schlüssels
- Homomorph (hier nicht behandelt)
 - Erlaubt es, Berechnungen mit verschlüsselten Werten auszuführen, ohne die Werte zu entschlüsseln
 - Oft auch nur einzelne Operationen möglich (→ additiv-homomorph)

$$E(a, K^+) \oplus E(b, K^+) = E(a + b, K^+)$$

$$E(a, K^+) \otimes E(b, K^+) = E(a \cdot b, K^+)$$

ARTEN VON VERSCHLÜSSELUNGSVERFAHREN

	Symmetrisch	Asymmetrisch
Schlüssellänge [bit]	üblicherweise 128 oder 256	üblicherweise ab 2048
Geschwindigkeit	schnell	langsam
Schlüsselverteilung	Sicherer Kanal erforderlich	Öffentlicher Schlüssel kann einfach verteilt werden
Verwendung	Verschlüsselung großer Datenmengen	Authentifizierung, digitale Signaturen, Schlüsselmanagement
Beispiele	DES, 3DES, AES	RSA, Diffie-Hellman, ElGamal

- Identifikation
 - Feststellen der Identität
 - Beispiel: Eingabe Username
 - Authentifikation
 - Sicherstellen der Identität
 - Beispiel: Eingabe Passwort
 - Autorisierung
 - Erteilen von Zugriffsrechten
 - Beispiel: Aufgrund Rollen oder Mitgliedschaften
- Typische Abhängigkeiten
 - Authentifikation benötigt Identifikation
 - Autorisierung benötigt Authentifikation
 - Abhängigkeiten sind nicht zwangsläufig
 - Identifikation ohne Authentifikation
 - Gesichtserkennung
 - Autorisierung ohne Authentifikation
 - Wohnungsschlüssel

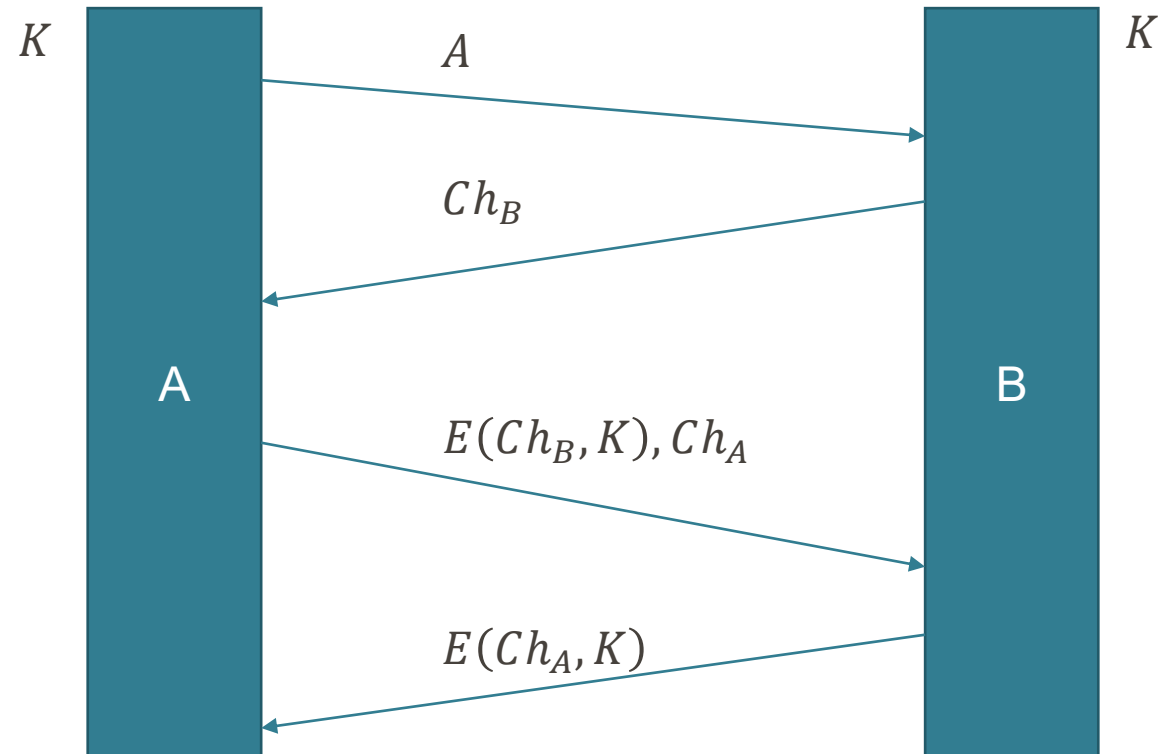
- Authentifizierung und Nachrichtenintegrität sind nicht trennbar
 - Authentizität ohne Integrität: Nachricht könnte verändert worden sein
 - Integrität ohne Authentizität: Nachricht könnte von jemand anderem kommen
- Typische Vorgehensweise
 1. Sicheren Kanal mit gegenseitiger Authentifizierung
 2. Geheimen (symmetrischen) Sitzungsschlüssel aushandeln
 3. Geheimen Sitzungsschlüssel verwenden, um Vertraulichkeit und Integrität zu sichern

AUTHENTIFIZIERUNG MIT SYMMETRISCHEN VERFAHREN

- Challenge-Response Protokoll

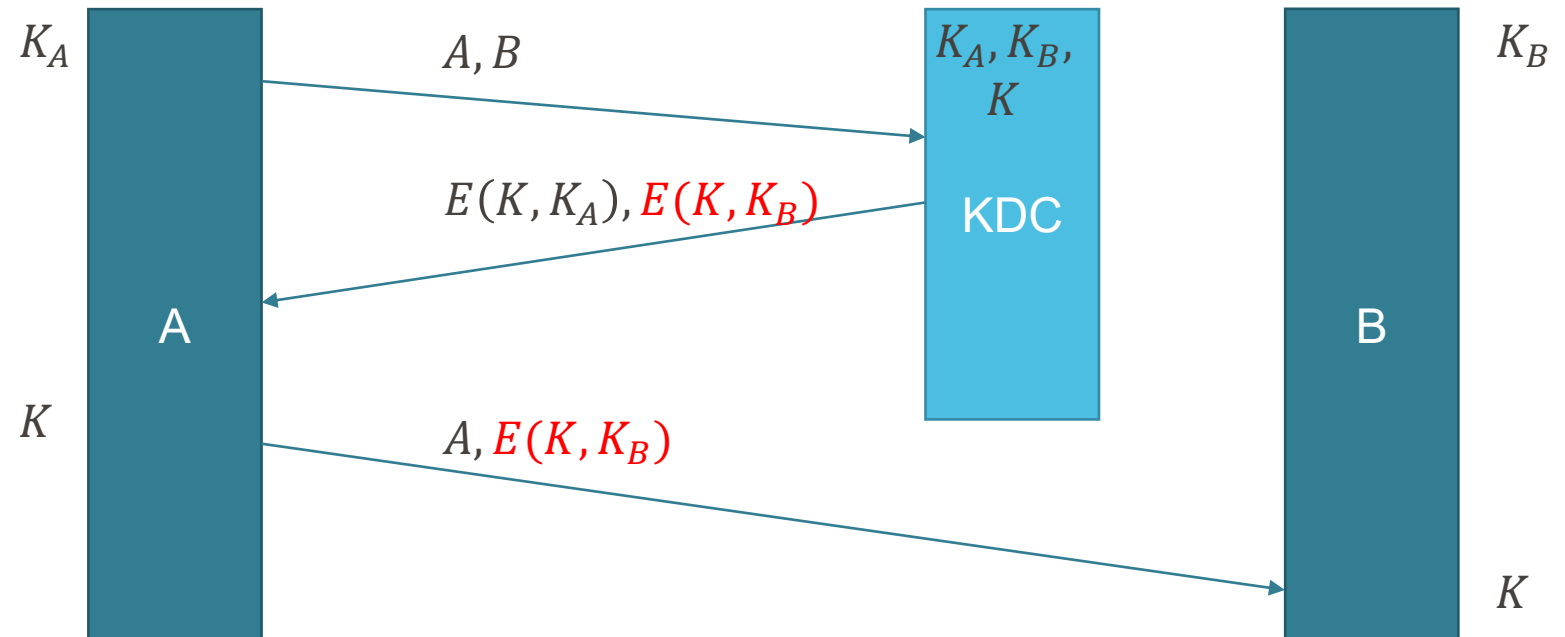
1. Kommunikationswunsch enthält Identität A
2. Challenge Ch_B (e.g. Zufallszahl) gestellt von B
3. Antwort enthält $E(Ch_B, K) \rightarrow A$ kennt gemeinsamen Schlüssel
4. Analog für andere Richtung

- Problem: Schlüssel für jede Kombination muss vorher bekannt sein



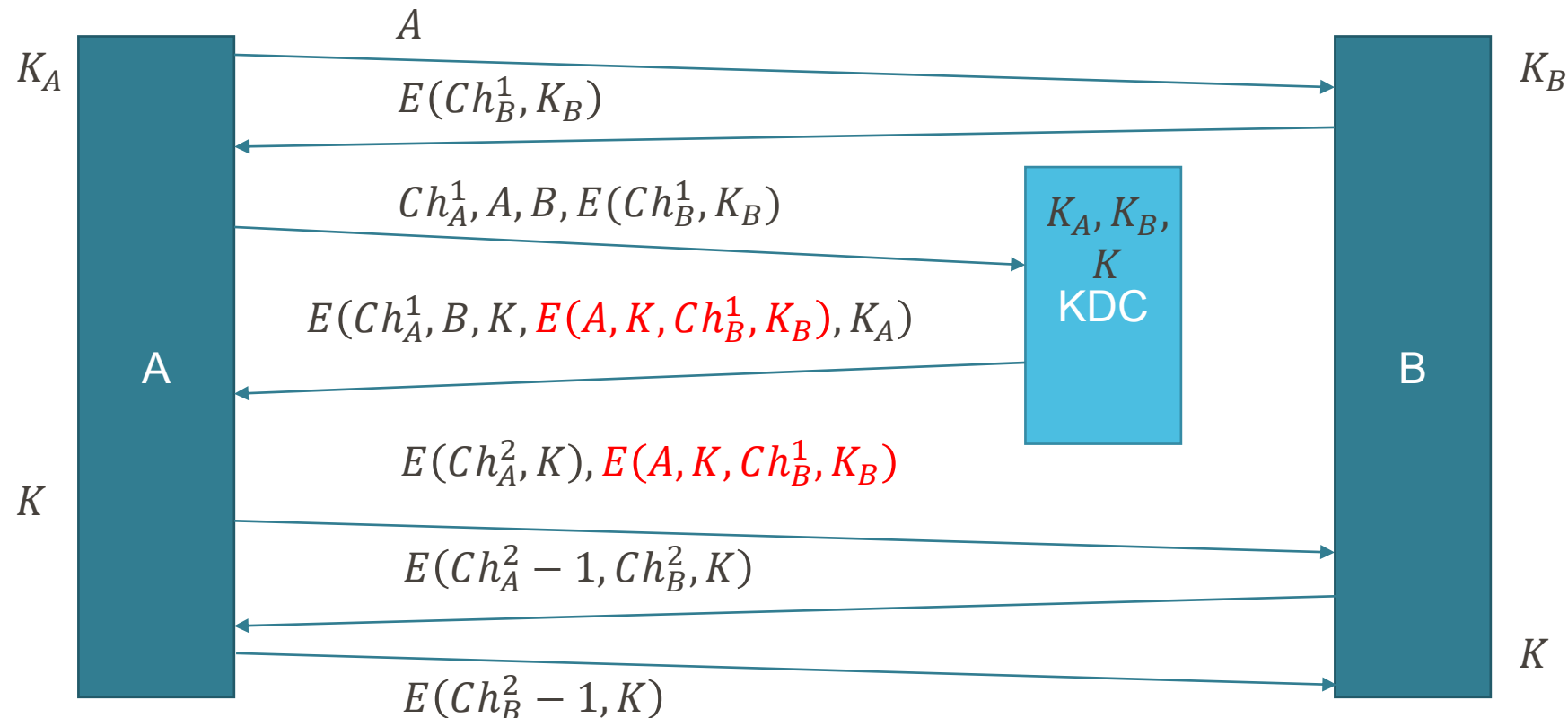
AUTHENTIFIZIERUNG MIT SYMMETRISCHEN VERFAHREN

- Einführung Tresor
 - Key Distribution Center (KDC)
 - Kennt/generiert Schlüssel für alle Kommunikationspaare
 - Teilnehmer kennen nur Schlüssel zur Kommunikation mit Tresor
- Tresor erstellt Ticket $E(K, K_B)$
 - Nur von B entschlüsselbar
- Problem: Ticket kann wiederholt werden



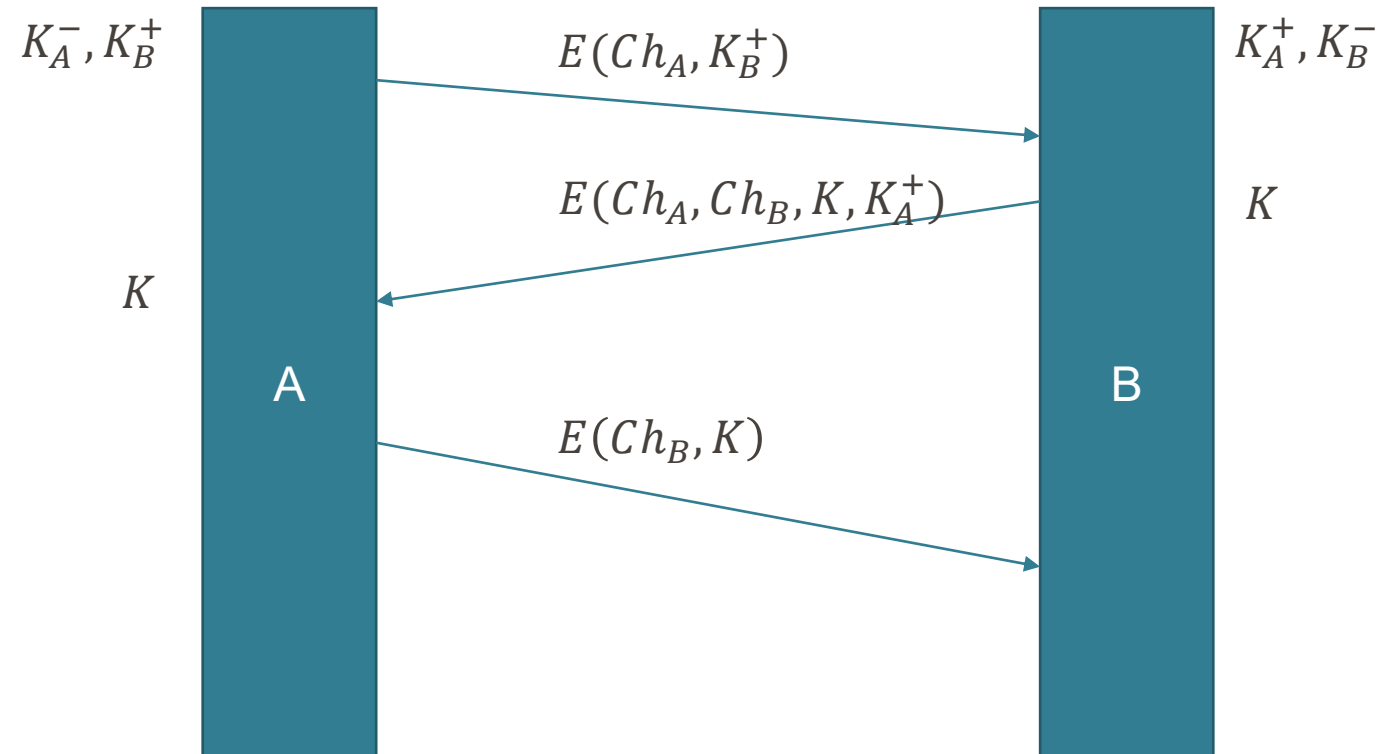
AUTHENTIFIZIERUNG MIT SYMMETRISCHEN VERFAHREN

- Weiterentwicklung: Needham-Schroeder-Protokoll
 - Needham, Schröder, 1978
 - Basis für Kerberos

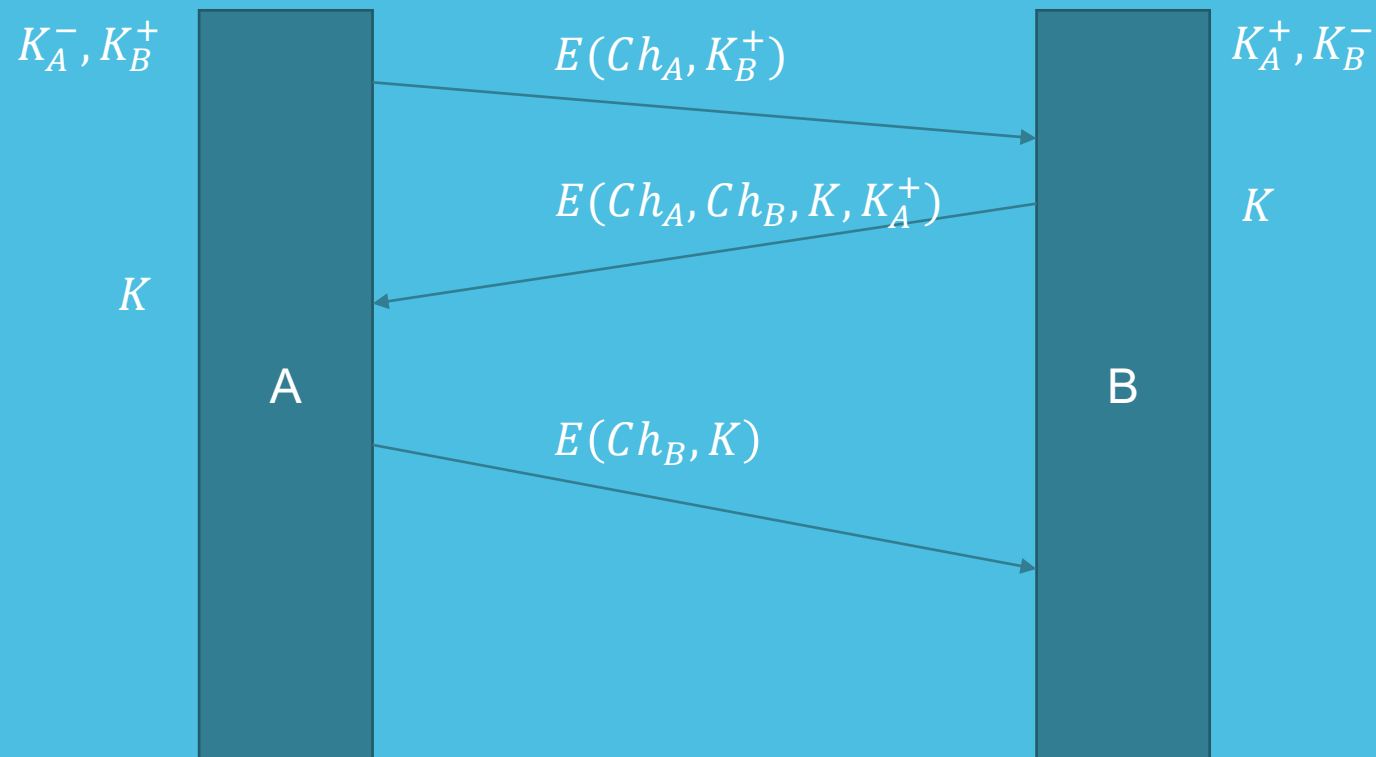


AUTHENTIFIZIERUNG MIT ASYMMETRISCHEN VERFAHREN

- Kein KDC erforderlich
 - K_A^+ ist öffentlicher Schlüssel, allen bekannt
 - K_A^- ist privater Schlüssel, nur A bekannt
 - Symmetrischer Sitzungsschlüssel K wird generiert, kurzlebig
- Erfordert klare Zuordnung von Kommunikationspartnern zu öffentlichen Schlüsseln



WOHER KÖNNTE B K_A^+ KENNEN?



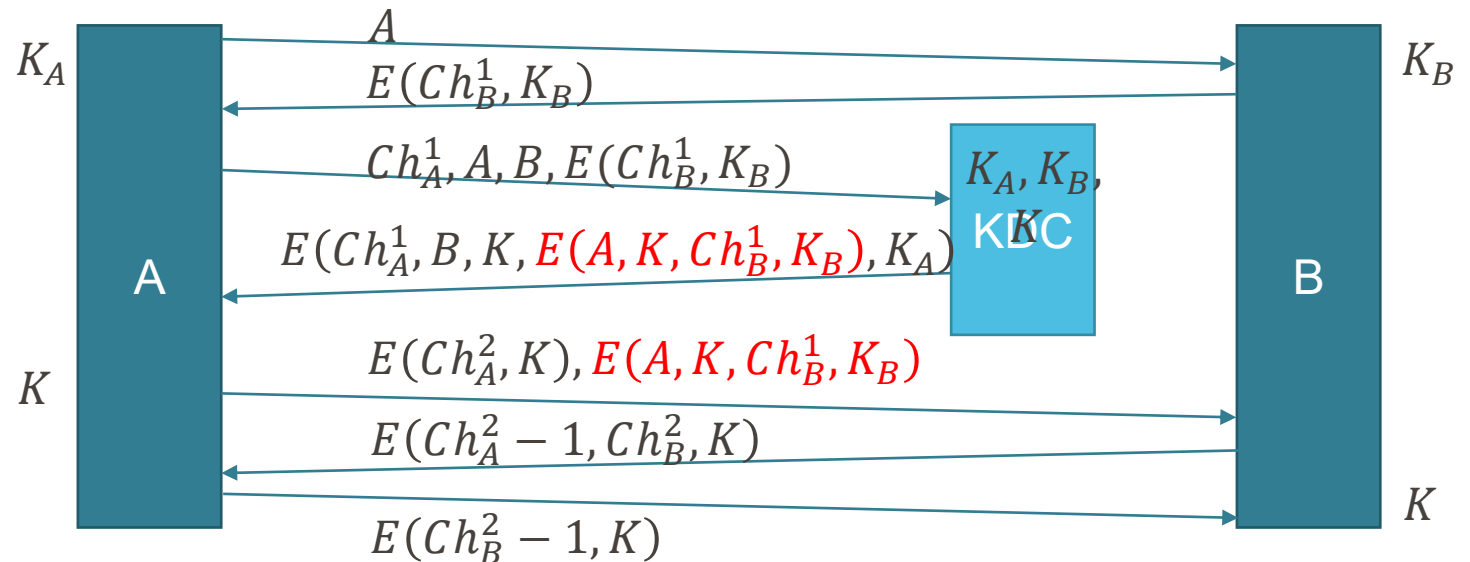
- Bis hierher: Austausch von öffentlichen Schlüsseln
 - A kennt öffentlichen Schlüssel von B
 - A schickt eigenen öffentlichen Schlüssel
- **Problem:** Woher wissen die Kommunikationspartner, wozu der Schlüssel gehört?


- Ziel: Sichere Verwaltung von Schlüsseln
 - Erstellen von Schlüsseln
 - Verteilen von Schlüsseln
 - Schlüssel ungültig machen (Revocation)
- Symmetrische Schlüssel → Key Distribution Center
- Asymmetrische Schlüssel → Public Key Infrastructure

SCHLÜSSELVERWALTUNG

Kerberos

- Ursprung MIT
- Basiert auf Needham-Schroeder-Protokoll (symmetrische Variante)
 - Erweiterung um Zeitstempel
 - Token haben nur begrenzte Gültigkeit
- Aufteilung KDC
 - Authentication Server
 - Ticket Granting Server
- Häufiger Einsatz
 - Microsoft Active Directory
 - OSF DCE
 - ...



- Zertifikat = Zuordnung öffentlicher Schlüssel → Person, Dienst, Organisation, etc.
 - Plus zusätzlicher Metadaten
- Problem: Woher weiß man, ob die Zuordnung stimmt?
- Zertifikatsketten
 - Zertifikat enthält neben Informationen zur Identität noch Fingerabdruck des Ausstellers (Certificate Authority, CA)
 - Ausstellendes Zertifikat signiert das Zertifikat mit dazugehörigem Schlüssel 
 - Aussteller ist für die Gültigkeit der Zuordnung öffentlicher Schlüssel → Person etc. verantwortlich
 - Sperrlisten (Revocation List) ermöglichen Rückruf von Zertifikaten
 - Zertifikat der CA selbst wieder von anderer CA ausgestellt (Vertrauenskette)
- Root CA stellt sich selbst Zertifikat aus, muss vom Client vertraut werden

- Digitaler Fingerabdruck über Dokumente/Nachrichten etc. („Message Digest“)
- Hash-Funktion H
 - $h = H(m)$
 - Länge der Nachricht m beliebig
 - Länge des Hashwertes h fest (z.B. 256 Bit)
- Annahmen
 - Berechnung von H ist effizient und eindeutig
 - Umkehrung ist schwierig (Einwegfunktion)
 - Kleine Änderungen an der Nachricht führen zu gänzlich anderen Hashwerten

KRYPTOGRAPHISCHE HASH-FUNKTIONEN

Secure Hash Algorithm

- SHA-1
 - 160 Bit Hashwert
 - Rechenintensive Angriffe bekannt
- SHA-2
 - Weiterentwicklung von SHA-1 (ab 2001)
 - Verschiedene Hashwert-Längen: SHA-224, SHA-256, SHA-384, SHA-512
 - Gilt noch als sicher (insb. SHA-512)
- SHA-3 (Keccak)
 - Bertoni, Daemon, Peeters, Van Assche
 - 2012 als Gewinner einer NIST-Ausschreibung gekürt
 - Gleiche Hashwert-Längen wie SHA-2

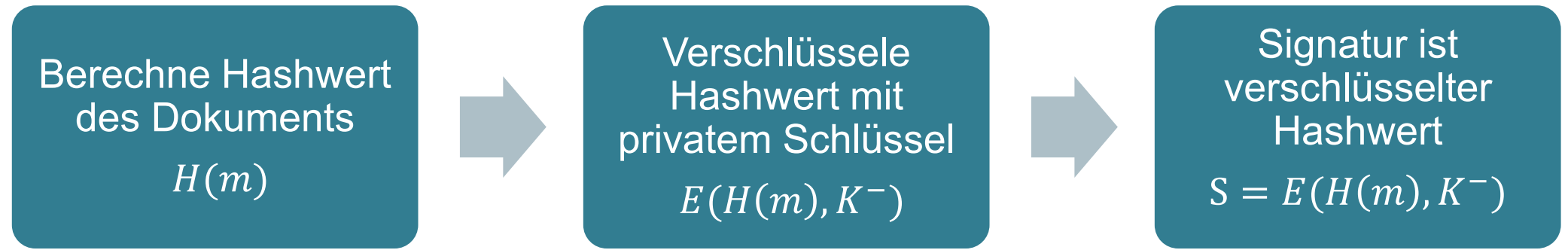
DIGITALE SIGNATUREN

- Unterschrift unter digitale Inhalte
 - Gebunden an unterschriebenes Dokument
 - Nicht (leicht) zu fälschen
- Kombination aus
 - Kryptografischer Hash-Funktion
 - Asymmetrischer Verschlüsselung
 - Public-Key Infrastruktur
- Wirksames Mittel zum Erreichen von Schutzzielen
 - Integrität
 - Authentizität
 - Nichtabstreitbarkeit
- Kein Schutz der Vertraulichkeit

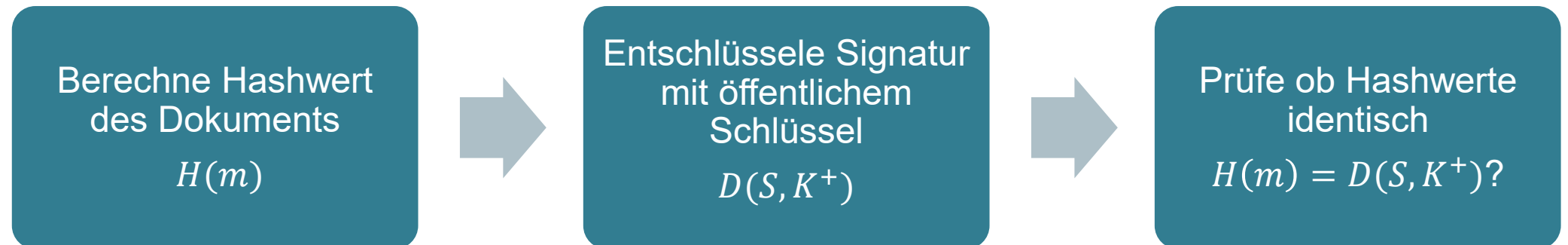


[Bild: Wikipedia, CC BY-SA, Tilt u]

Berechnung der Signatur



Prüfung der Signatur



QUALIFIZIERTE SIGNATUREN IN DEUTSCHLAND

Anbieterliste nach Bundesnetzagentur (Feb 2023)

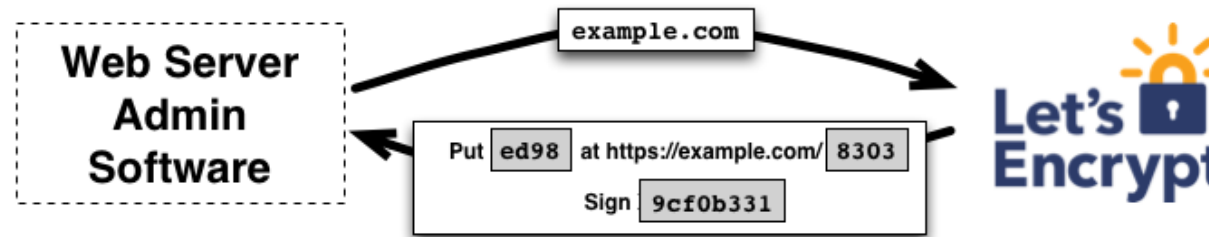
- Atos Information Technology GmbH
- Bank-Verlag GmbH
- Bundesnotarkammer
- Deutsche Telekom AG
- DGN Deutsches Gesundheitsnetzwerk Service GmbH
- Deutsche Post AG
- D-Trust GmbH
- medesign GmbH

Gesetzliche Voraussetzung: Identifikation nach
eIDAS-Verordnung bzw. Vertrauensdienstegesetz

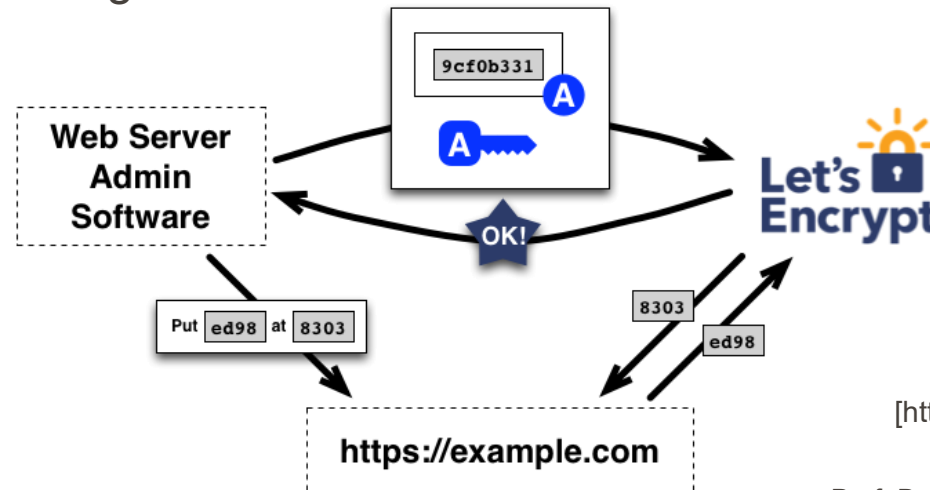
FREIE ZERTIFIZIERUNGSSTELLEN

Let's Encrypt

- Zertifikate üblicherweise kostenpflichtig → hemmt Sicherheit im Web
- Mehrstufiges Verfahren für kostenfreie Zertifikate
 1. Administrator fordert Zertifikat für Website, erhält Challenge an bestimmten Ort bestimmten Inhalt zu posten



2. Let's Encrypt prüft Challenge und erteilt Zertifikat



[<https://letsencrypt.org/de/how-it-works/>]

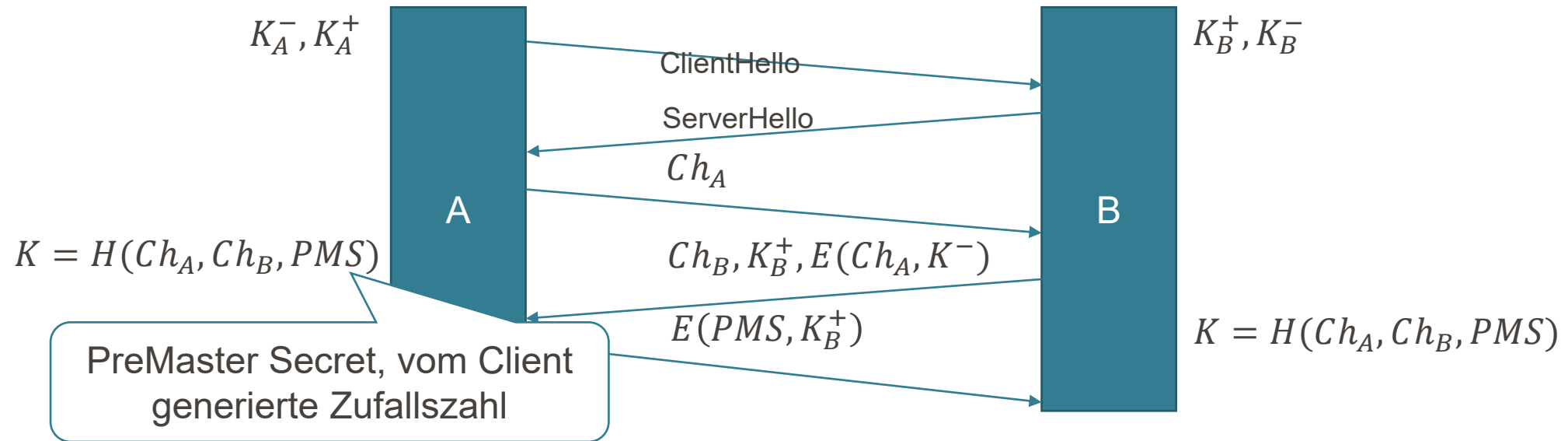
- Öffentlicher Schlüssel des Zertifikatinhabers
- Name des Inhabers (Distinguished Name)
 - Common Name (CN): Gebräuchlicher Name
 - Organization (O): Firma oder Organisation
 - Organizational Unit (OU): Abteilung oder Organisationseinheit
 - Locality (L): Stadt, Sitz der Organisation
 - State (ST): Staat, Provinz, Gegend
 - Country (C): ISO Ländercode
- Name der ausstellenden CA (Distinguished Name), Thumbprint
- Gültigkeitszeitraum
- Verwendete Algorithmen
- Erweiterungen (bspw. Verwendungszweck)

TRANSPORT LAYER SECURITY (TLS)

- Nachfolger von Secure Socket Layer (SSL)
 - TLS 1.0 ~ SSL 3.1
 - Aktuell TLS 1.3: RFC 8446 (2018), TLS 1.2 noch unterstützt
- Sicherheit auf Transportebene
 - TLS dekoriert Transportschicht → Benötigt zuverlässige Transportschicht als Basis
 - Transparenz für alle höheren Anwendungsprotokolle (SMTP, IIOP, ...)
 - Optional beidseitige Authentifikation (mutual TLS)
 - Hohe praktische Bedeutung
- Ursprung: Netscape, Ersatz für unsichere 40 Bit Verschlüsselung
- Verbreitete Implementierungen
 - OpenSSL, GnuTLS, LibreSSL, BoringSSL, ...

TRANSPORT LAYER SECURITY (TLS)

Ablauf



- Unterstützung verschiedener Verschlüsselungsverfahren
 - Diffie-Hellman, seit TLS 1.3 kein RSA mehr
- Erste Nachrichten verhandeln verwendete Algorithmen
 - ClientHello: TLS-Version, unterstützte Algorithmen
 - ServerHello: Gewählte Algorithmen

TRANSPORT LAYER SECURITY (TLS)

Zertifikate

- Übertragung von öffentlichen Schlüsseln als X.509 Zertifikate
- Client führt mehrere Prüfungen durch:
 - Zertifikat auf Kommunikationspartner ausgestellt?
 - CN ist Host-Name (DNS-Name)
 - CN ist IP-Adresse
 - IP-Adresse oder DNS-Name als Subject Alternative Name (SAN)
 - ...
 - Alle Zertifikate in Zertifikatskette noch gültig?
 - Zertifikatskette endet bei vertrauenswürdiger Root CA?

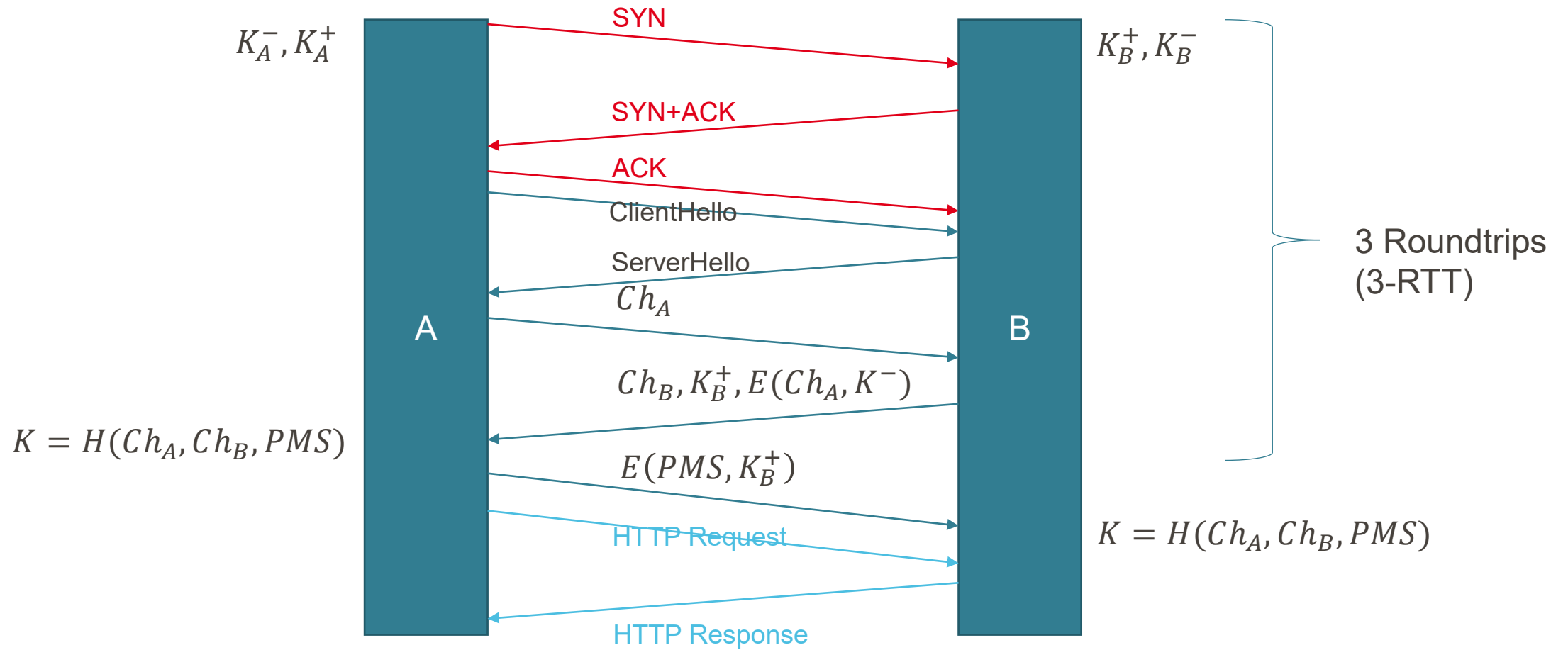
TRANSPORT LAYER SECURITY (TLS)

Forward Secrecy („Folgenlosigkeit“)

- Problem: Angreifer könnte irgendwann doch Schlüssel knacken
- Lösung
 - Häufiger Wechsel des Sitzungsschlüssels
 - Verwende gemeinsamen Schlüssel nur, um jeweils aktuelle Schlüssel zu signieren

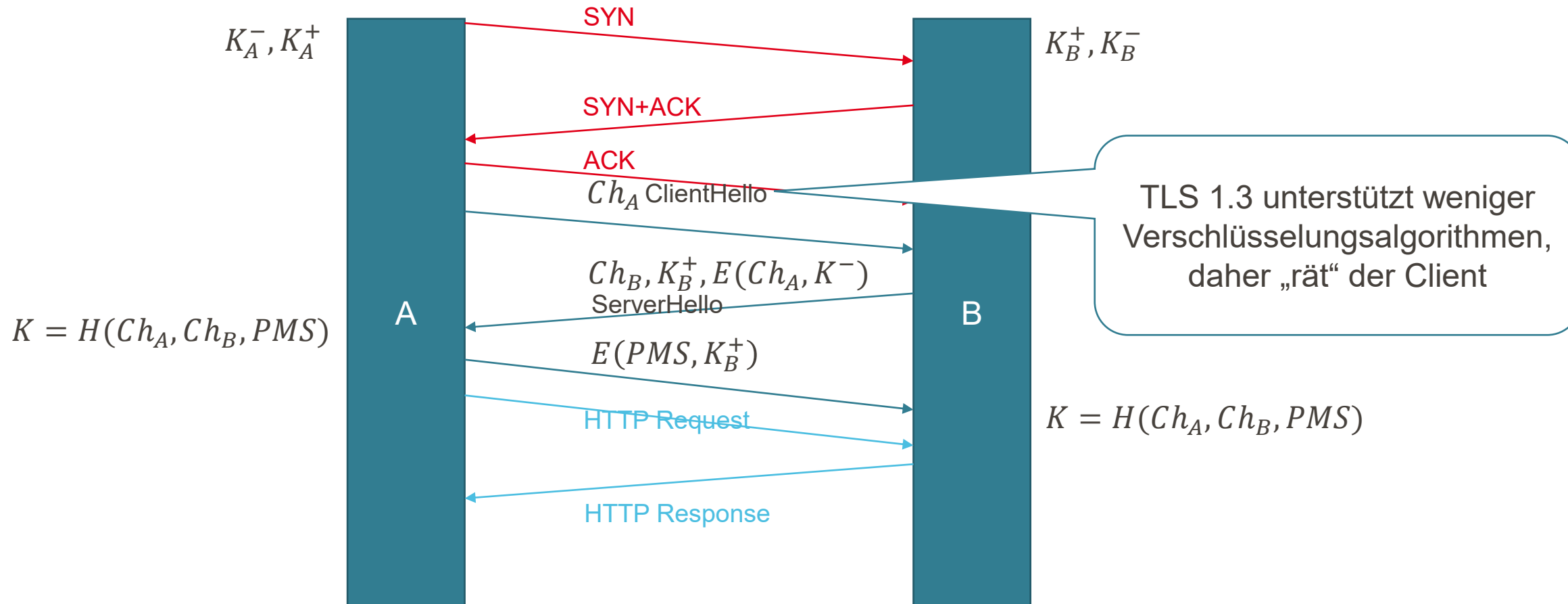
TLS 1.2 VIA TCP

3-RTT



TLS 1.3 VIA TCP

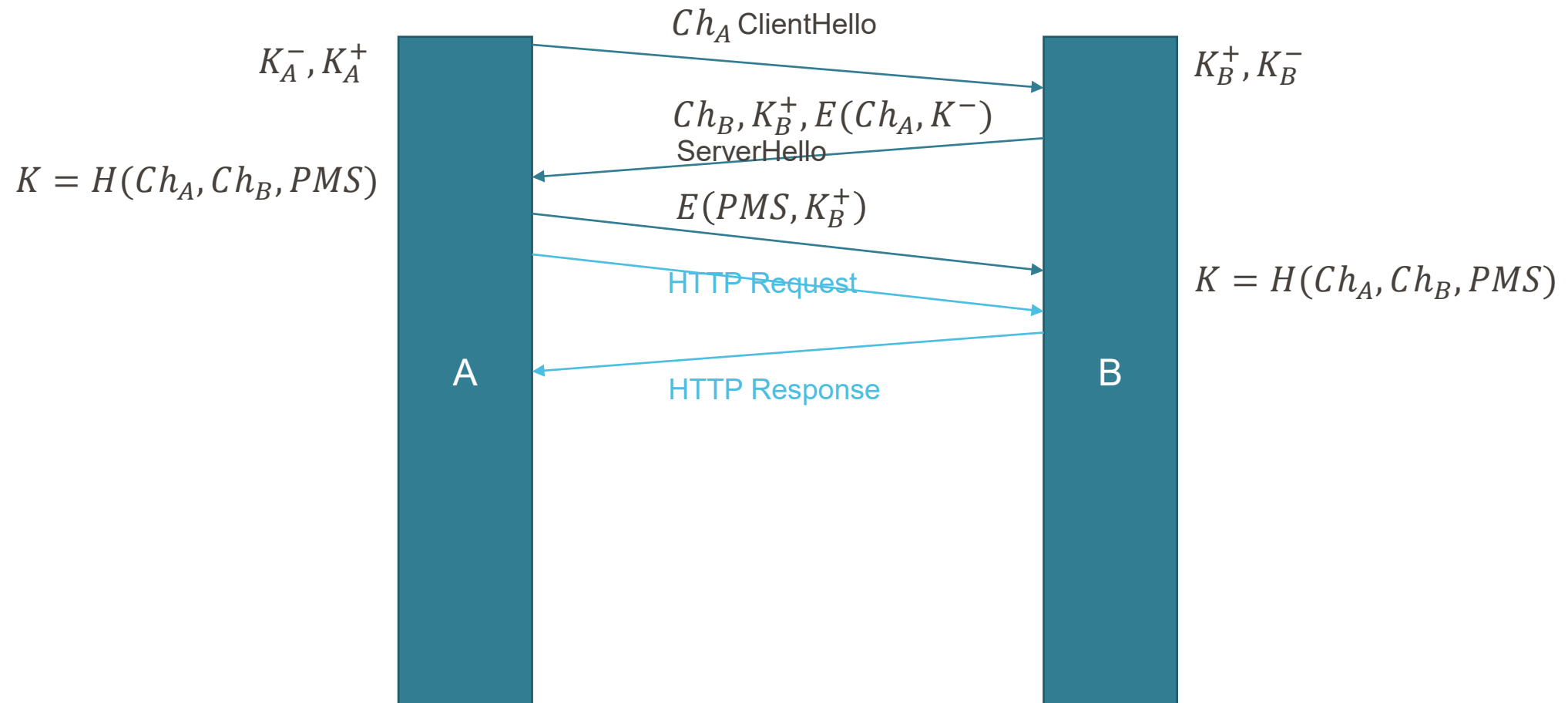
2-RTT



Sollte der Server das Verfahren nicht unterstützen, greift Verhandlung wie bei TLS 1.2

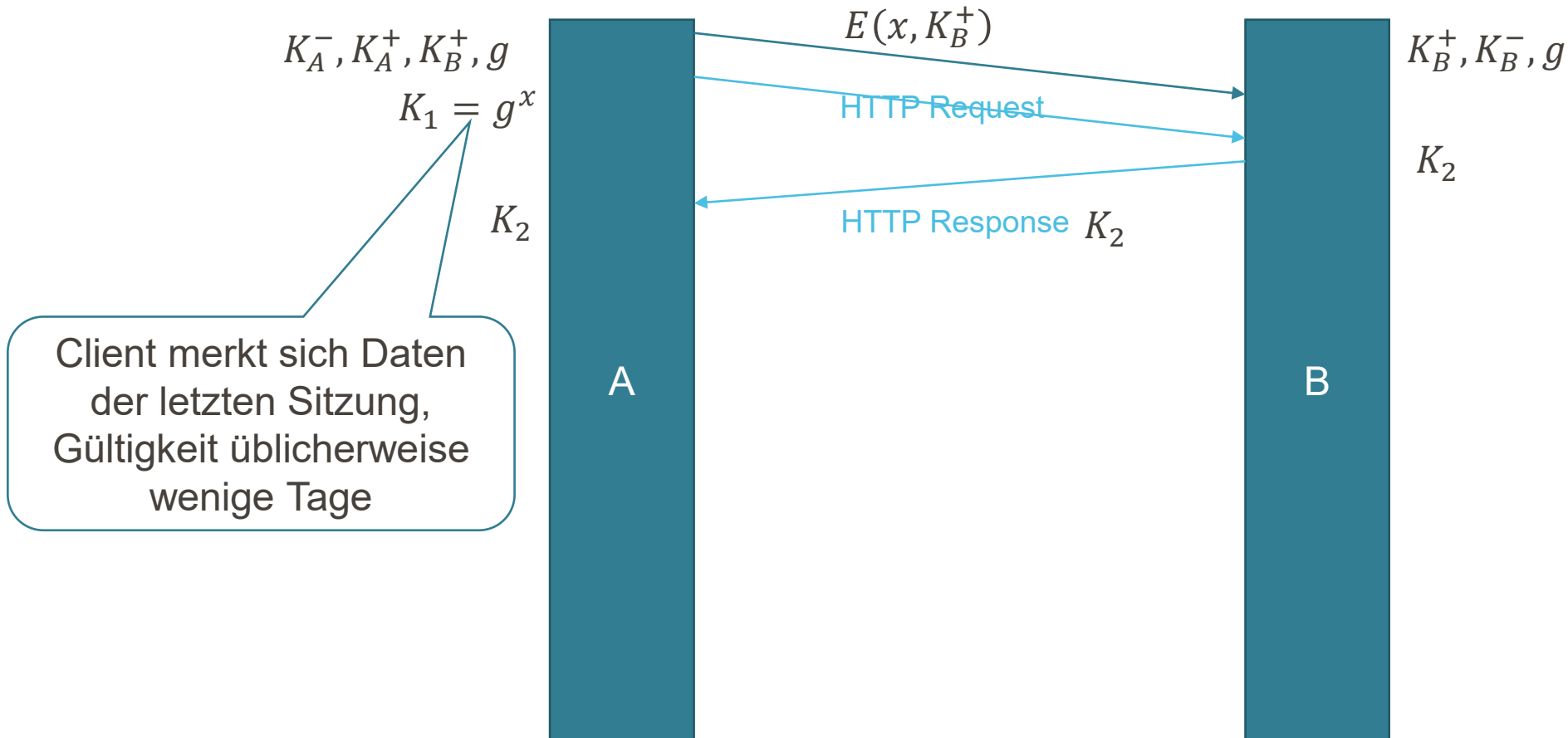
TLS 1.3 VIA QUIC

1-RTT



TLS 1.3 VIA QUIC

0-RTT



Verfahren abhängig von Verschlüsselungsalgorithmus, vor allem verwendet bei Diffie-Hellman

- Transportsicherheit durch kryptografische Verfahren
 - Asymmetrische Verschlüsselung
 - X.509 Zertifikate
 - Relativ komplexe Verfahren, um Sitzungsschlüssel auszuhandeln
 - Unterstützte Verschlüsselungsalgorithmen ändern sich häufig (alle paar Jahre)
- Große Unterschiede in der Latenz
 - HTTP via TCP/TLS 1.2: 3 Roundtrips
 - HTTP via QUIC/TLS 1.3: 0 Roundtrips im common case
 - Client sendet sofort Nutzdaten



- Welche Arten von Verschlüsselungsverfahren gibt es?
- Was ist eine digitale Signatur?
- Was ist ein KDC?
- Erläutern Sie die Authentifizierung von Kommunikationspartnern nach dem Needham-Schröder Protokoll!
- Was ist eine PKI?
- Warum kann ein Client einem vom Server ausgestelltem Zertifikat trauen?
- Warum ist der Sitzungsaufbau mit TLS 1.3 oft schneller als mit TLS 1.2?
- Unter welchen Bedingungen kann mit TLS 1.3 ein 0-RTT Sitzungsaufbau zustande kommen?