Westfälische Hochschule
Gelsenkirchen Bocholt Recklinghausen
University of Applied Sciences

Transport Layer Security (TLS)
Secure Socket Layer (SSL)

Prof. Dr. (TU NN)
Norbert Pohlmann

Institut für Internet-Sicherheit – if(is)
Westfälische Hochschule, Gelsenkirchen
http://www.internet-sicherheit.de

TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

TLS/SSL-Technologie
→ Inhalt

- **Ziele und Ergebnisse der Vorlesung**
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Ziele und Ergebnisse der Vorlesung → TLS/SSL

- Gutes **Verständnis für** den Sinn und Zweck der **TLS/SSL-Technologie** und deren Anwendungsmöglichkeiten.

- Erlangen der **Kenntnisse** über die **Architektur**, Aufgaben, **Prinzipien** und **Sicherheitsmechanismen** der TLS/SSL-Technologie

- Gewinnen von **praktische**n **Erfahrungen** über die TLS/SSL-Technologie mit Hilfe eines Protokollmittschnittes

TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- **Einleitung**
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Einleitung
→ Idee von TLS/SSL

- Da das **Internet offen** ist und die **Angriffsmöglichkeiten** sowie **Angriffswahrscheinlichkeiten** sehr groß sind, ist die Nutzung einer ***verschlüsselten* und *integritätsgesicherten Kommunikation*** zwischen Client und Server von besonderer Bedeutung.

- Sehr viele **Cyber-Sicherheitsaspekte** im Web, wie Eingabe von Passwörtern und Kreditkarten-Informationen haben mit der Einrichtung einer **vertrauenswürdigen Verbindung zwischen Client und Server** zu tun.

- **Der vorherrschende Ansatz für die Transportverschlüsselung** ist die Verwendung von **TLS (Transport Layer Security)** / SSL (Secure Socket Layer) – **TLS/SSL**.

- **TLS/SSL** ist ein **anwendungsunabhängiges Cyber-Sicherheitsprotokoll,** das logisch **auf einem Transportprotokoll** aufsetzt.

Einleitung
→ Cyber-Sicherheitsfunktionen von TLS/SSL

- **Authentifikation** von Server und Client unter Verwendung von *asymmetrischen Verschlüsselungsverfahren* und *elektronischen Zertifikaten*.

- **Vertrauliche Client-to-Server** (Ende-zu-Ende)**Datenübertragung** mit Hilfe *symmetrischer Verschlüsselungsverfahren* unter der Nutzung eines *gemeinsamen Sitzungsschlüssels*.

- **Sicherstellung der Integrität und Authentizität** der transportierten Daten unter Verwendung des *HMAC-Verfahrens*.

- TLS/SSL bietet auch die Komprimierung der Daten an.

Einleitung
→ Geschichte

- Die **Idee kam von Netscape**, die die erste Version von SSL 1994 veröffentlichte.
- 1999 wurde SSL von der **IETF als Standard** festgelegt und umbenannt zu Transport Layer Security (TLS).
- Da aber heute noch im Sprachgebrauch SSL fest verankert ist, wird im Weiteren immer von **TLS/SSL** gesprochen.

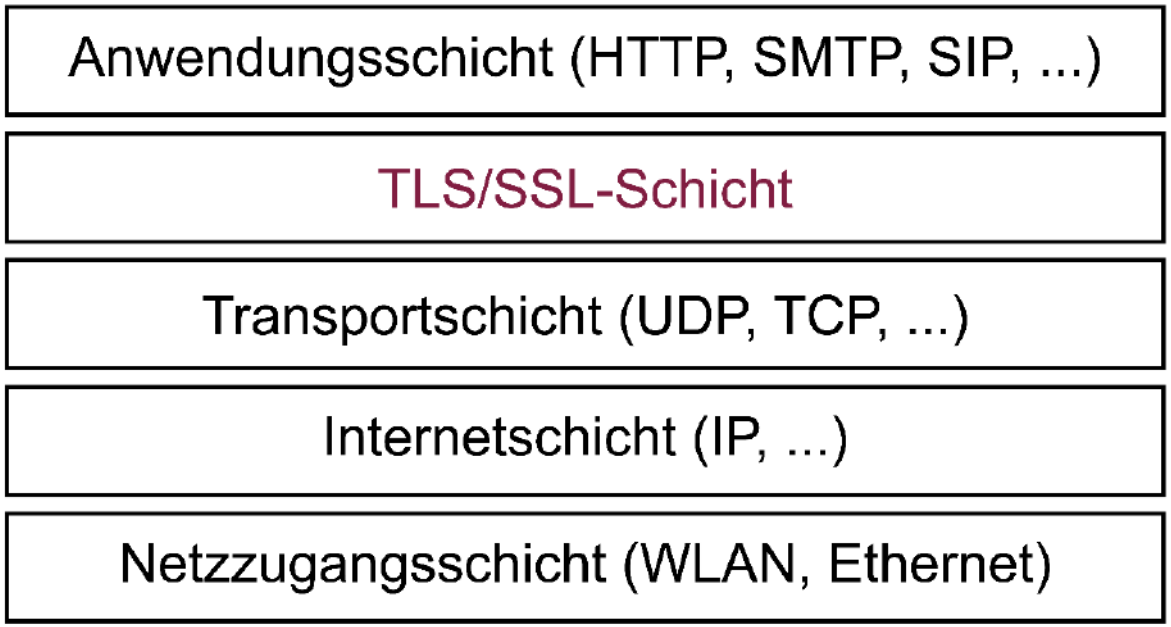
Praktische Relevanz von TLS/SSL

- Da **TLS/SSL in allen wichtigen Internet-Technologien** wie Browsern, Web-Servern, E-Mail-Servern usw. eingebunden ist, wird TLS/SSL faktisch als **Standard für die Transportverschlüsselung** verwendet.

TLS/SSL-Technologie
→ Inhalt

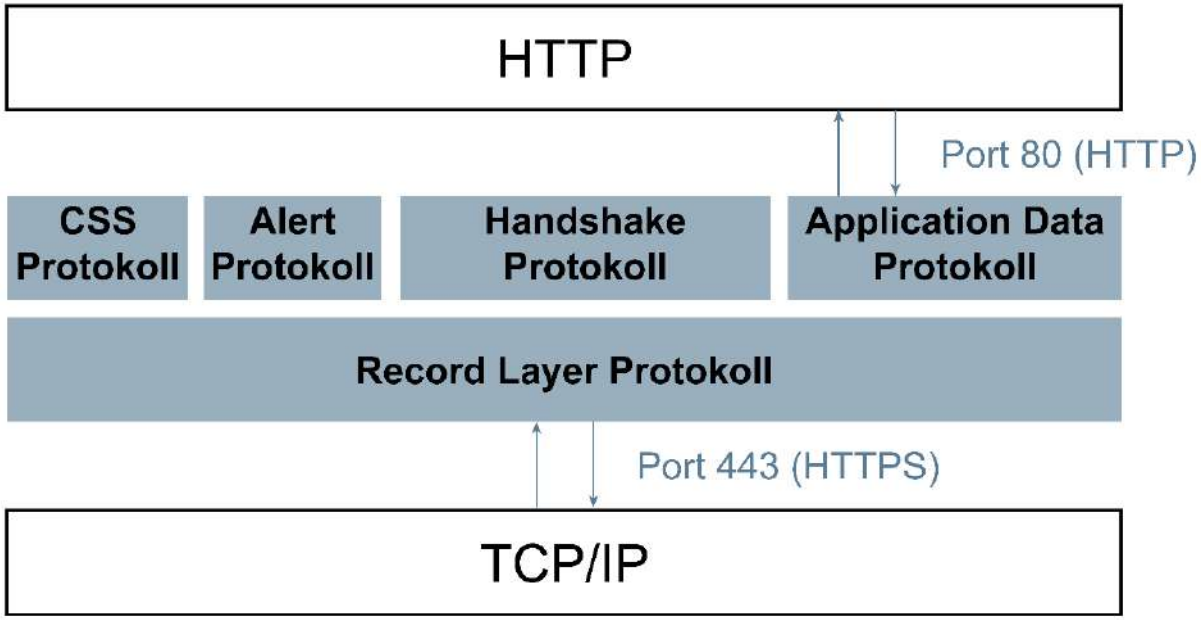
- Ziele und Ergebnisse der Vorlesung
- Einleitung
- **Architektur und Protokolle**
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Einbindung in die
→ Kommunikationsarchitektur



- **TLS/SSL** kann eine Vielzahl **höherer Anwendungsprotokolle** unterstützen, wie HTTP, SMTP, SIP, IMAP, FTP, Telnet.
- Die genauen **Sicherheitseigenschaften** des TLS/SSL-Kanals werden **bei der Einrichtung** der *verschlüsselten* und *integritätsgesicherten Kommunikation* zwischen Client und Server **festgelegt**.
- Sicherheitseigenschaften sind: **Authentifikation von Client und Server** sowie **Integrität**, **Authentizität** und **Vertraulichkeit** der **Transportdaten**.

Architekturaufbau
→ der TLS/SSL-Schicht



- **Die TLS/SSL-Schicht besteht aus zwei Teil-Schichten:**
  - höhere Schicht mit den **TLS/SSL-Teil-Protokollen** (CCS, Alert, Handshake, Application)
  - Record-Schicht mit dem **Record Layer-Protokoll**

- Der Client kann **durch die Wahl des Ports entscheiden**, ob die Kommunikation **TLS/SSL-gesichert sein soll** oder im Klartext:
  - Port 80: http-Kommunikation im Klartext
  - Port 443: http-Kommunikation TLS/SSL-gesichert

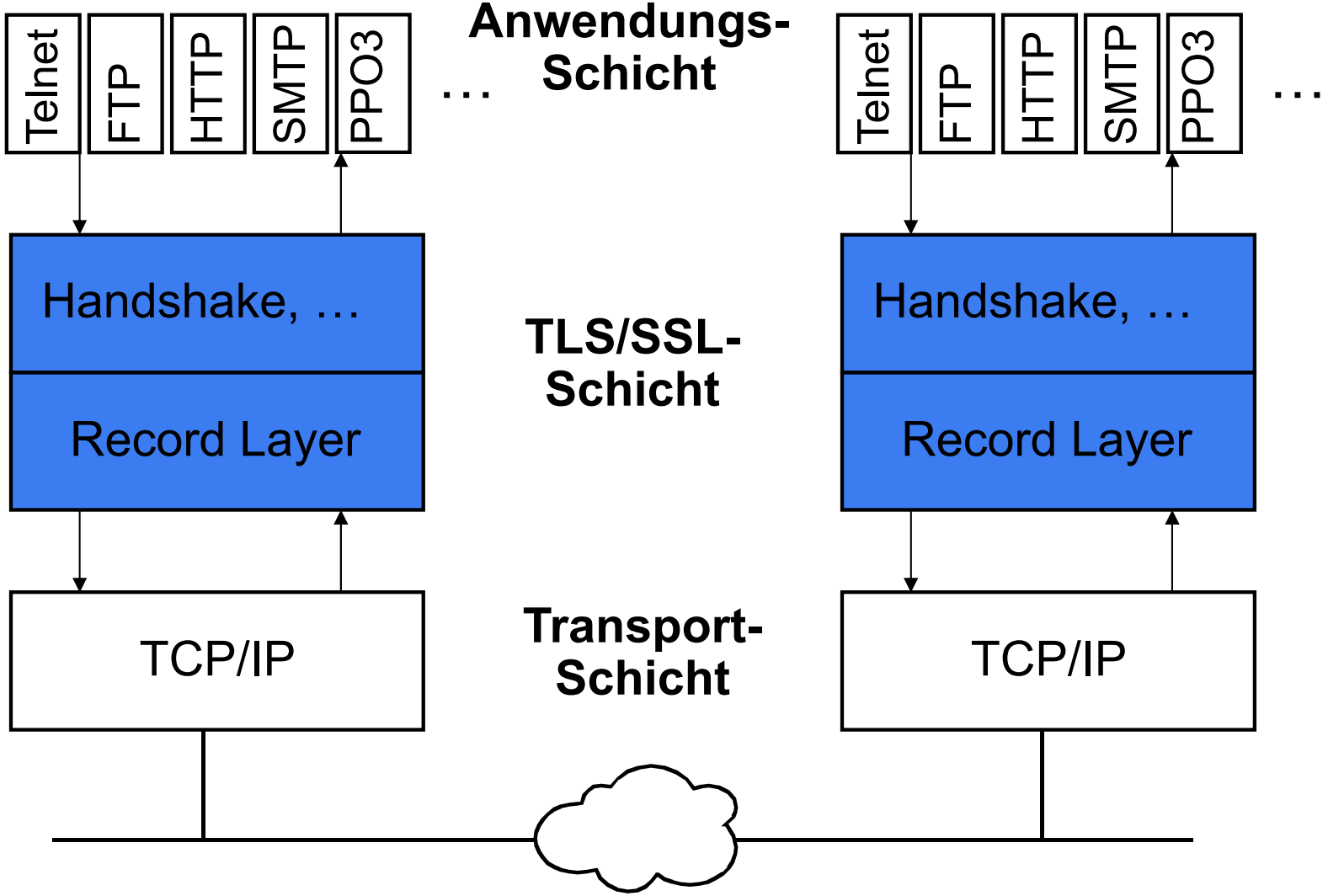
Anwendung
→ der TLS/SSL-Schicht

Kommunikationsanwendung	Port-Nummer	TLS/SSL Port-Nummer
Hypertext Transfer Protocol - HTTP	80	443
Simple Mail Transfer Protocol - SMTP	25	465
Internet Message Access Protocol - IMAP	143	993
Session Initiation Protocol – SIP	5060	5061
File Transfer Protocol – FTP	20, 21	989, 990
Teletype Network – TELNET	23	992
…	…	..

- Die Umsetzung der TLS/SSL-Schicht funktioniert so, dass für die Anwendungsprotokolle eine **weitere Port-Nummer** als **Transportadresse** definiert wurde, die genutzt wird, wenn die Kommunikation zu diesen **Anwendungen TLS/SSL-gesichert** umgesetzt werden soll.

- Anhand der **Port-Nummer** lässt sich erkennen, um welches ursprüngliche **Anwendungsprotokoll** es sich bei den übertragenen Daten handelt. (z.B. 443 → 80)

TLS/SSL-Protokoll
→ Schichteneinordnung (1/2)



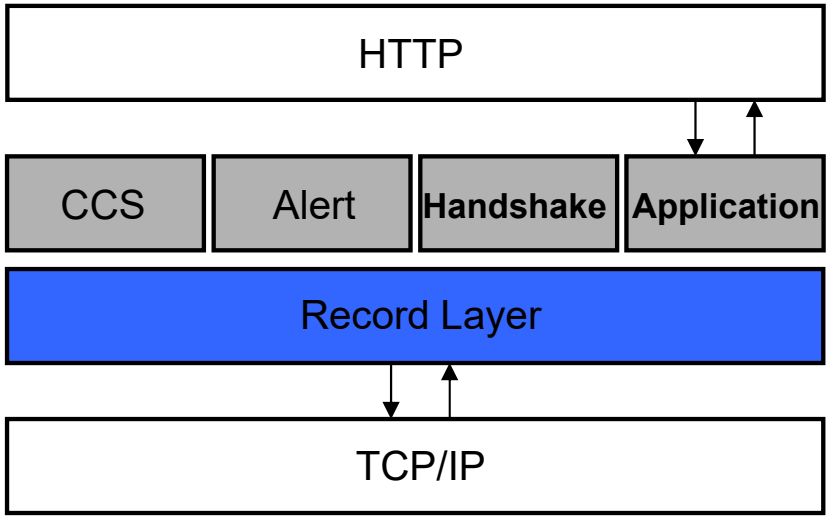
- Die TLS/SSL-Schicht befindet sich zwischen der Transport- und Anwendungsschicht.

TLS/SSL-Protokoll
→ Schichteneinordnung (2/2)

- **TLS/SSL** übernimmt **zusätzlich** die Aufgaben **der Sitzungs- und Präsentationsschicht** (Schichten 5 und 6) des ISO/OSI-Modells.
- Ein wesentlicher Vorteil der Sitzungsschicht gegenüber der Transportschicht besteht darin, dass **Zustandsinformationen über einen längeren Zeitraum** und über verschiedene Einzelverbindungen hinweg gespeichert und für die Verwaltung genutzt werden können.
- Für das zustandslose HTTP-Protokoll, das für jeden Zugriff auf eine Webseite eine neue TCP-Verbindung aufbauen kann, bedeutet das, dass mehrere solcher Verbindungen zu einer **TLS/SSL-Sitzung** gebündelt und damit **effizienter** als die jeweiligen Einzelverbindungen **verwaltet** werden können.
- Die TLS/SSL-Protokolle sind erweiterbar und flexibel, um Zukunftssicherheit vor allem bei den Verschlüsselungsalgorithmen zu gewährleisten.
- **TLS/SSL arbeitet transparent**, so dass es leicht eingesetzt werden kann, um Anwendungsprotokollen/-diensten, ohne eigene Cyber-Sicherheitsmechanismen, **vertrauenswürdige Verbindungen** zur Verfügung zu stellen.

TLS/SSL – Record Layer Protokoll
→ Aufgaben

- Das Record Layer Protokoll **leitet die Klartext-Anwendungsdaten aus der Anwendungsschicht verschlüsselt an die Transportschicht weiter**.
- Das Record Layer-Protokoll bietet zwei verschiedene Cyber-Sicherheitsdienste, die zusammen oder einzeln genutzt werden können:
  - **Client-to-Server-Verschlüsselung** zwischen den beiden Transport-Endpunkten
  - Sicherung der Nachrichten-Integrität und -Authentizität
- Zu den Aufgaben des Record-Protokolls gehören
  - die **Fragmentierung** der Klartext-Anwendungsdaten der Anwendungsschicht
  - **Kompression** der resultierenden Fragmente
  - **Berechnung von HMACs** über die (komprimierten)Fragmente
  - **Verschlüsselung** der komprimierte Fragment mit HMAC



TLS/SSL – Record Layer Protokoll
→ Aufbau

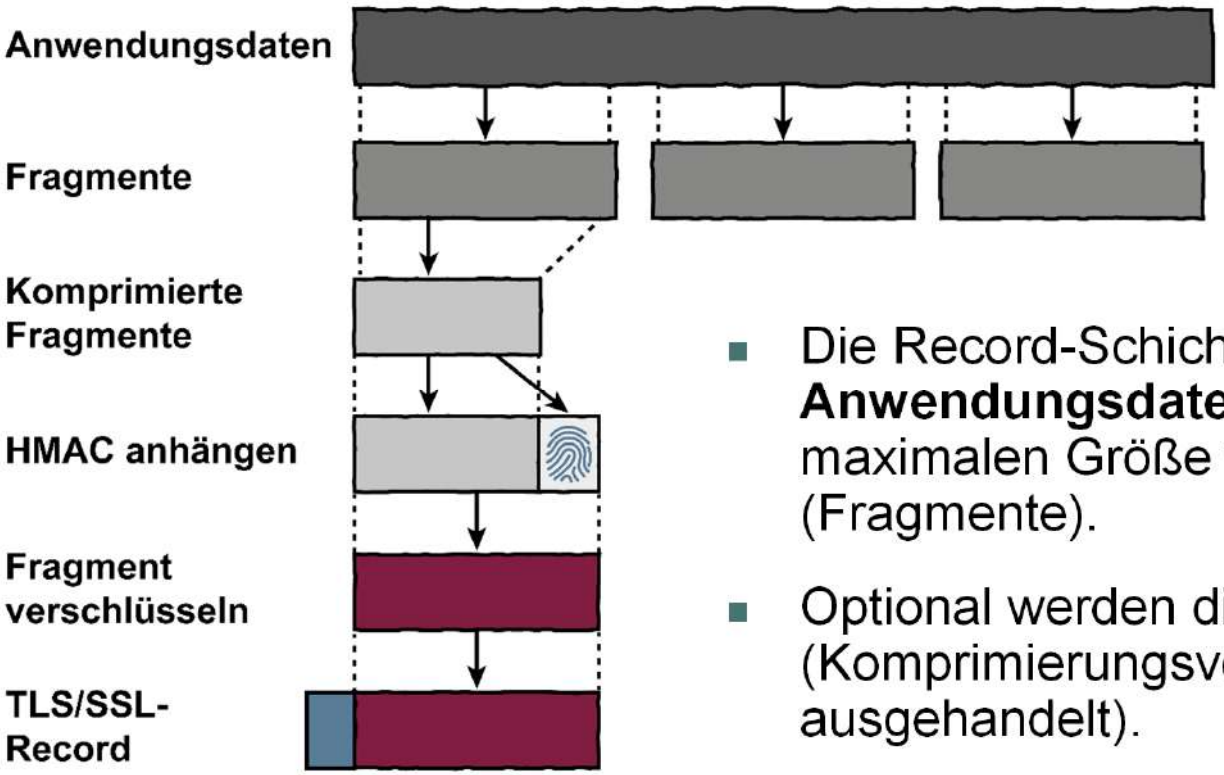
- Gesamtgröße: 5 Byte

0	1	2	3	5
Type	**Version Major**	**Version Minor**	**Length**	

- **Type** (1 Byte) : Nummer des “höheren” TLS/SSL-Protokolls
  - Change Cipher Spec: 20
  - Alert: 21
  - Handshake: 22
  - Application Data: 23
- **Version Major** (1 Byte) : Hauptnummer der Version
- **Version Minor** (1 Byte) : Unterversion
- **Length** (2 Byte) : Länge der Nutzdaten in Byte.

  Maximalwert darf nicht grösser sein als 2^{14} + 2.048 (16.384 + 2.048 Byte).

TLS/SSL – Record Layer Protocol
→ Ablauf



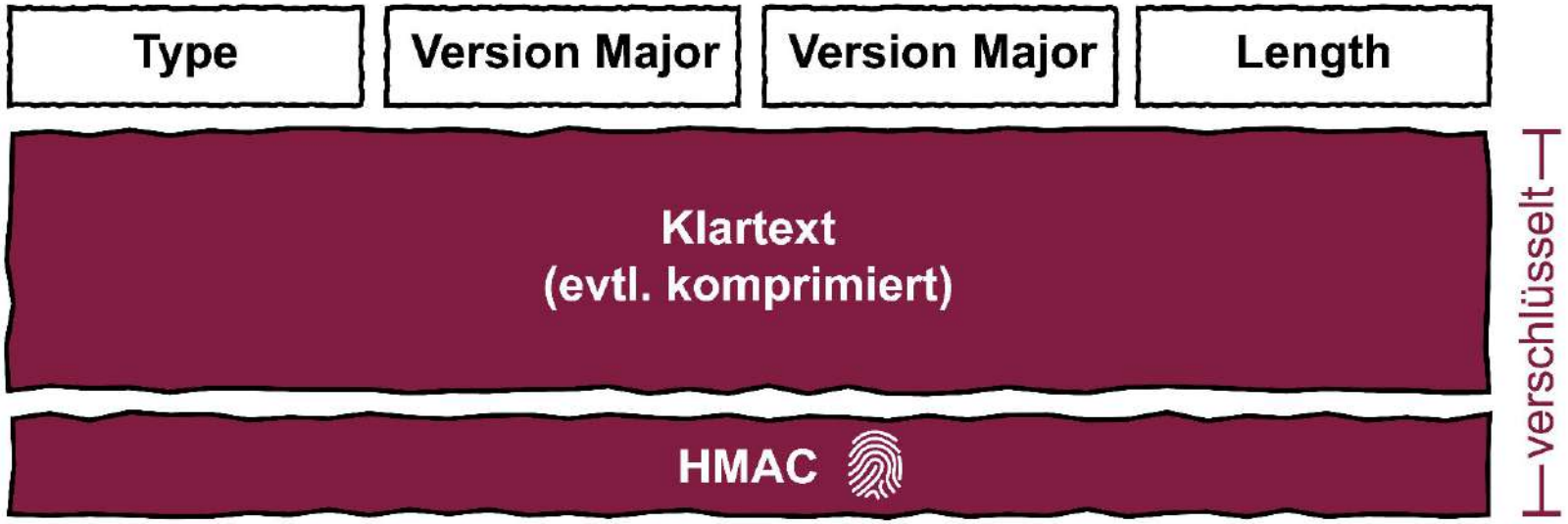
- Die Record-Schicht hat die Aufgabe, **Anwendungsdaten** in TLS/SSL-Records mit einer maximalen Größe von 2^{14} Byte zu **fragmentieren** (Fragmente).
- Optional werden die **Fragmente** dann **komprimiert** (Komprimierungsverfahren wird beim Handshake ausgehandelt).
- Diese **(komprimierten)Fragmente** + Sequenznummer werden dann mit einer HMAC-Funktion **gehasht**.
- Die **Fragmente** und der **HMAC** werden dann **verschlüsselt**.
- Aus dem verschlüsselten Fragment wird ein **TLS/SSL-Record gebildet**.

TLS/SSL – Record Layer Protocol
→ HMAC Berechnung

HMAC = KH (HMAC-Key,
SeqNum || Compressed.type || Compressed.version ||
Compressed.length || Compressed.fragment **)**

KH = “Keyed-Hashing for Message Authentication”-Verfahren; HMAC-Verfahren

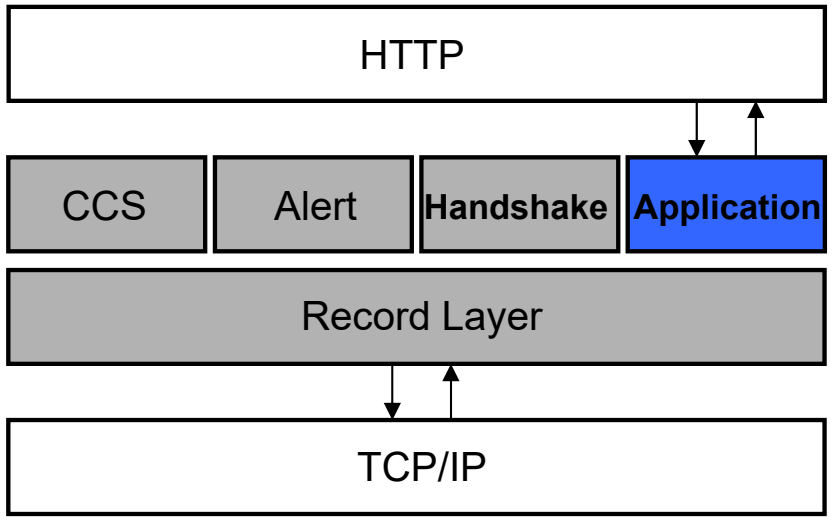
TLS/SSL – Record Layer Protocol
→ Verschlüsselte Daten



- Die eigentlichen Nutzdaten der Protokolle werden bei TLS/SSL verschlüsselt über das Application Data Protokoll übertragen.
- Der komplette Record-Layer-Header ist ungeschützt und somit lesbar.
- Zudem kann beim Handshake-Protokoll die Art der Handshake-Nachricht ausgelesen werden.

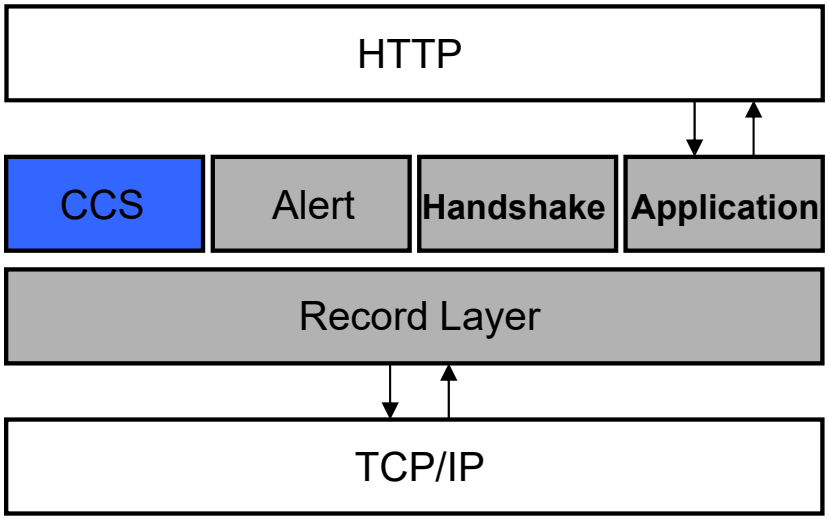
TLS/SSL – Application Data Protokoll
→ Aufgaben

- Das **Application Data Protokoll** ist definiert, um die Anwendungsdaten transparent, d.h. ohne Betrachtung des Inhalts, durchreichen zu können.
- Die anderen Protokolle oberhalb des Record Protokolls werden auf ihren Inhalt hin untersucht, dies ist bei den Daten des Application Data Protokoll nicht der Fall.
- Die Daten werden entsprechend der Sicherheitsparameter (aus dem Handshake Protokoll) fragmentiert, komprimiert, gegen Verfälschung geschützt und verschlüsselt.
- Gesamtgröße: Variabel



TLS/SSL – ChangeCipherSpec (CCS)
→ Aufgaben

- Das **ChangeCipherSpec** Protokoll wird bei **Änderung des kryptografischen Algorithmus bzw. Parametern** genutzt .
- Es enthält nur eine Meldung bestehend aus einem Byte mit dem Wert 1.
- CCS bewirkt, dass der Empfänger die während des Handshakes ausgehandelten Parameter für die aktive Sitzung übernimmt.
- Wird eine vorhandene Sitzung reaktiviert, erfolgt die Meldung ChangeCipherSpec nach der Meldung ServerHelloDone.



TLS/SSL – Alert Protokoll
→ Aufgaben

- Das Alert Protokoll dient der **Signalisierung von besonderen Zuständen bzw. Problemen** wie Fehler oder Verbindungsabbruch.
- Protokollnachricht enthält nur 2 Felder: Sicherheitslevel und Fehlercode.
- Gesamtgröße: 2 Byte

0	1	2
Type	**Error**	

- Das erste Byte kann den Wert **warning**(1) oder **fatal**(2) haben und gibt den Grad der Fehlermeldung an.
- Wird der Wert **“fatal”** übertragen, **beendet TLS/SSL** sofort die Verbindung.
- Andere Verbindungen aus der Session bleiben bestehen, aber es kann keine neue Verbindung erzeugt werden.
- Das zweite Byte beschreibt den Fehler genauer.

TLS/SSL – Alert Protokoll
→ Fatal

Fatale Alertmeldungen:

Name	Inhalt
Bad_record_mac(20)	Falscher HMAC für Record
Record_overflow(22)	Record Länge ist größer als 2^14 + 2.048 Byte
Decompression_failure(30)	Dekomprimierungsfunktion erhält falschen Input
Handshake_failure(40)	Sender akzeptiert die Sicherheitsparameter nicht
Illegal_parameter(47)	Feld im Handshake war inkonsistent
unknown_ca(48)	Root-Zertifikat konnte nicht gefunden werden oder ist nicht unter den vertrauenswürdigen Zertifizierungsinstanzen
access_denied(49)	Zertifikat gültig, aber die Zugangskontrolle hat entschieden, keinen Zugriff zu gewähren
decode_error(50)	Länge der Nachricht falsch oder Nachricht konnte nicht decodiert werden, da ein Feld nicht korrekt ist
decrypt_error(51)	Beim Handshake ist die kryptografische Operation fehlgeschlagen; Sender war nicht fähig, die Signatur oder das Ende der Nachricht zu verifizieren
protocol_version(70)	Protokollversion wird nicht unterstützt
insufficient_security(71)	Server erwartet höhere Cipher Suite als der Client unterstützt
internal_error(80)	Interner Fehler unabhängig von Nutzern

TLS/SSL – Alert Protokoll
→ Warning

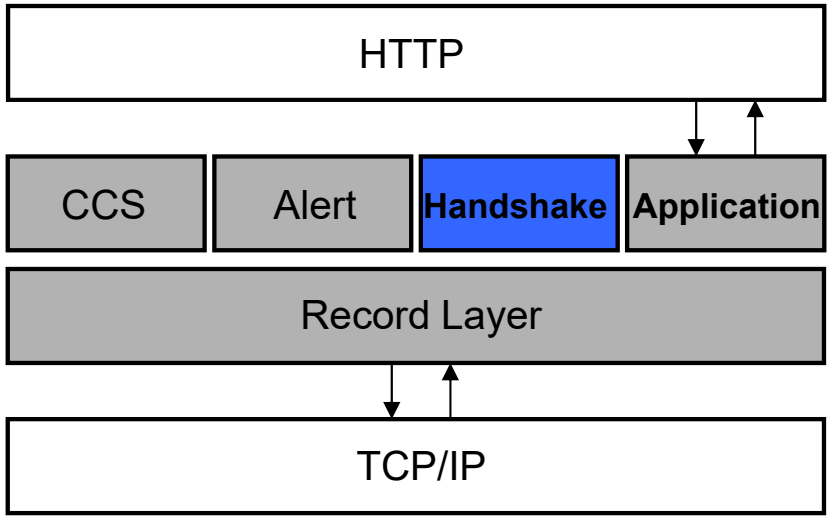
Warnungs Alertmeldungen:

Name	Inhalt
bad_certificate(42)	Zertifikat konnte nicht erfolgreich verifiziert werden
unsupported_certificate(43)	Zertifikat eines nicht unterstützten Types
certificate_revoked(44)	Zertifikat wurde vom „Signer“ gesperrt
certificate_expired(45)	Zertifikat ist abgelaufen
certificate_unknown(46)	Sonstige Zertifikatsprobleme
user_canceled(90)	Handshake von Nutzer abgebrochen
no_renegotiation(100)	Wenn nach dem ClientHello oder Hello die Parameter neu ausgehandelt werden sollen
unsupported_extension(110)	Versendet von Clients, die vom Server eine Erweiterung bekommen, die sie nicht verlangt haben

TLS/SSL – Handshake Protokoll
→ Aufgaben

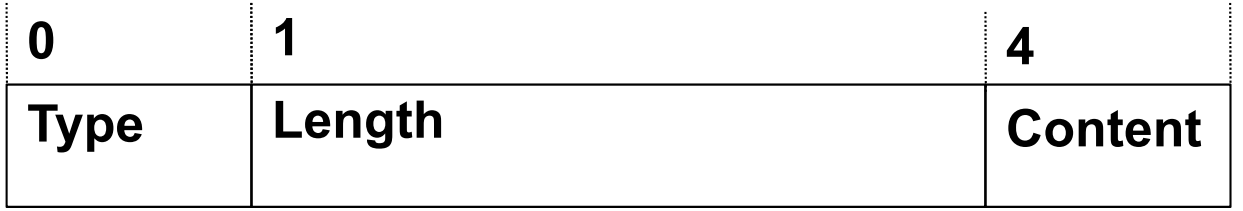
- Die Handshake-Nachrichten ermöglichen den **Aufbau einer TLS/SSL-Verbindung**.
- Dient u.a. zur
  - **Identifikation und Authentifizierung** der Kommunikationspartner, sowie
  - zum **Aushandeln kryptographischer Algorithmen, Schlüssel und Parameter**,

    die u.a. im TLS/SSL Record Layer Protokoll verwendet werden und zum Austausch benötigter geheimer Informationen.



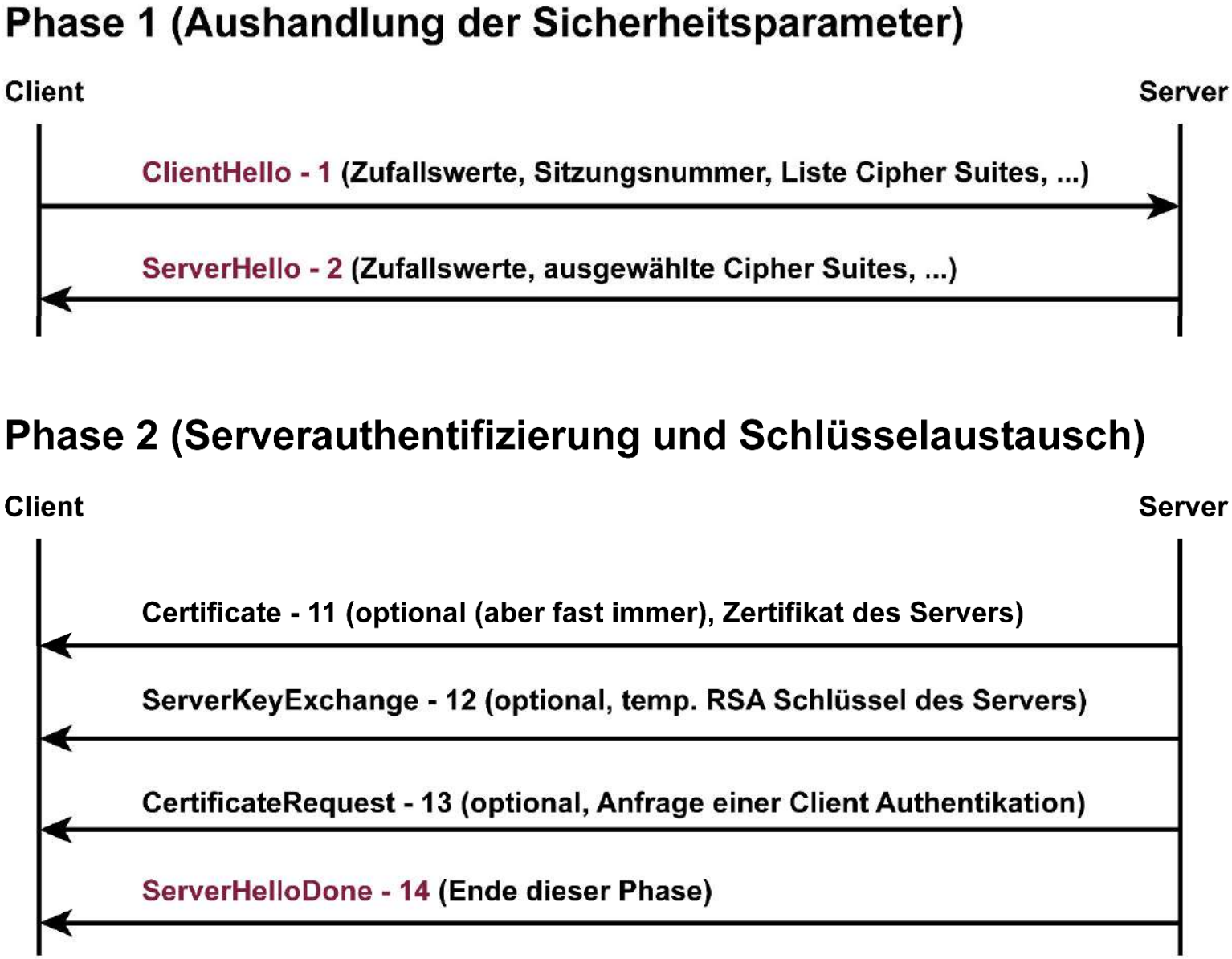
TLS/SSL – Handshake Protokoll
→ Aufbau

- Gesamtgröße: 5 Byte



- Type (1 Byte): Zeigt eine von 10 möglichen Nachrichten an (siehe Abb.)
- Length (3 Byte): Länge der Nachricht in Bytes
- Content (1 B.): Parameter, welche mit dieser Nachricht assoziiert sind

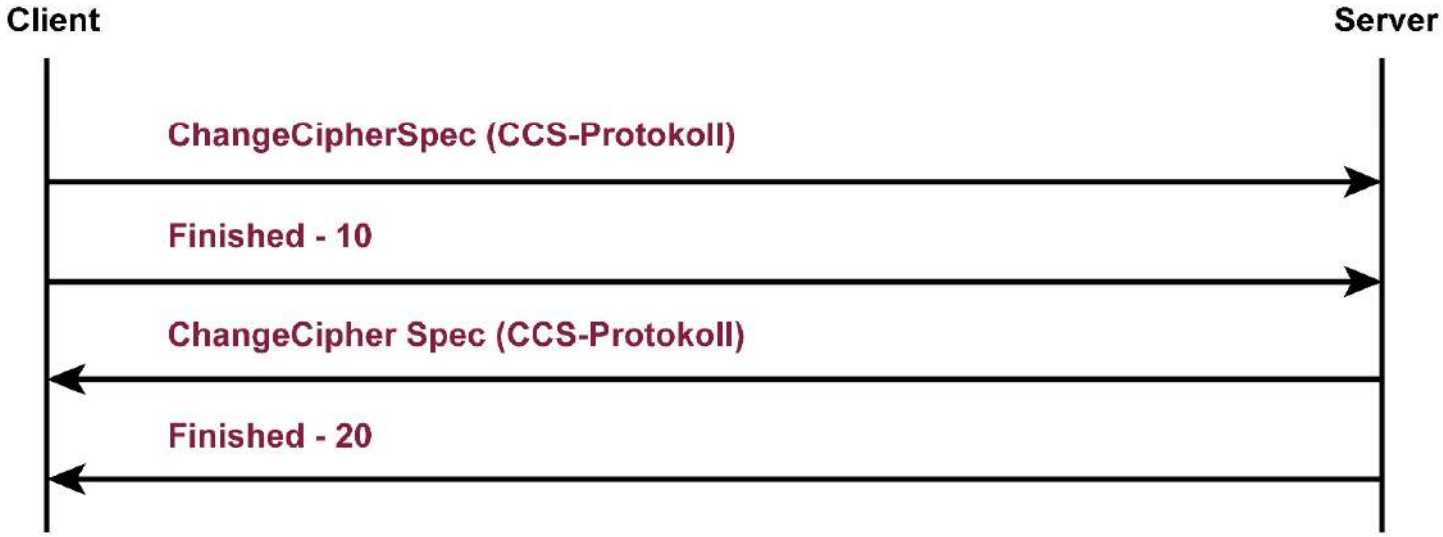
TLS/SSL – Handshake Protokoll
→ Nachrichten-Typen (1/2)



TLS/SSL – Handshake Protokoll
→ Nachrichten-Typen (2/2)

Phase 3 (Clientauthentifizierung und Schlüsselaustausch)

Phase 4 (Cipher Suite wird aktiviert und der Handshake beendet)



TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- **Protokollablauf: Prinzipien, Schritte und Phasen**
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Protokollablauf
→ Informationsaustausch (1/2)

- TLS/SSL ist ein **zustandsbehaftetes Cyber-Sicherheitsprotokoll**, mit dem **Sitzungen** zwischen Kommunikationspartnern **etabliert werden können**.
- Ein Client kann zu einem Zeitpunkt **mehrere** solche **Sitzungen** zum **gleichen oder zu verschiedenen Servern** unterhalten.
- TLS/SSL benutzt eine Session, um **Zustandsinformationen** über einen **längeren Zeitraum** zu speichern und nutzen zu können.
- Wie bei IPSec nutzt TLS/SSL für die **bidirektionale Verbindung** zwischen Client/Server zwei unterschiedliche Sitzungsschlüssel.
- Kryptographische Verfahren und Hashfunktion werden pro Sitzung ausgehandelt.
- TLS/SSL versucht, möglichst wenig geheime Informationen über das nicht vertrauenswürdige Transportsystem Internet zu übertragen.
- Daher werden lediglich Basisinformationen ausgetauscht, womit die beteiligten Kommunikationspartner (Client und Server) dezentral ihre Geheimnisse, wie die gemeinsamen Session Keys und HMAC-Schlüssel, berechnen.

Protokollablauf
→ Informationsaustausch (2/2)

- Eine TLS/SSL-Session ist eine „**Security-Assoziation**“ zwischen einem Client und einem Server.
- Sessions werden über das Handshake-Protokoll aufgebaut.
- Eine Session definiert eine **Menge von kryptographischen Sicherheitsparametern**, die gemeinsam über mehrere Verbindungen genutzt werden können.
- Der Sinn der Session besteht darin, nicht jedes Mal eine neue zeitaufwendige Verhandlung der Sicherheitsparameter auszuführen.

Sitzungs- und Verbindungskonzepte
→ Session Zustände

Ein Session-Zustand ist definiert über folgende Parameter:

- **Session-Identifier** Zufällige Bytesequenz, die vom Server erzeugt wird, um eine aktive oder wiederherstellbare Session zu identifizieren.
- ***Peer-Certificate*** X509.v3 Zertifikat des Clients (falls vorhanden).
- ***Compression-Method*** Kompressionsalgorithmus.
- ***Cipher-Spec*** Definiert den Verschlüsselungsalgorithmus sowie die One-Way-Hash-funktion für den HMAC. Es werden noch weitere Attribute, wie die HMAC-Länge, festgelegt.
- ***Master-Secret*** 48-Byte, die vom Client und Server benutzt werden, um die geheimen Schlüssel daraus^abzuleiten.
- ***Is-Resumable*** Mit diesem Flag wird angezeigt, ob diese Session für neue Verbindungen genutzt werden kann.

Sitzungs- und Verbindungskonzepte
→ TLS/SSL-Connection

- Die **TLS/SSL-Connection** ist ein **logischer Transportkanal** zwischen Client und Server.
- Es handelt sich bei TLS/SSL um eine Peer-to-Peer-Verbindung, die nur temporär genutzt wird.
- Die **TLS/SSL-Connection** ist immer mit einer **Session assoziiert**.
- Damit können **aus einer Session mehrere TLS/SSL-Connections** aufgebaut und parallel betrieben werden.

Sitzungs- und Verbindungskonzepte
→ Zustände einer TLS/SSL-Connection

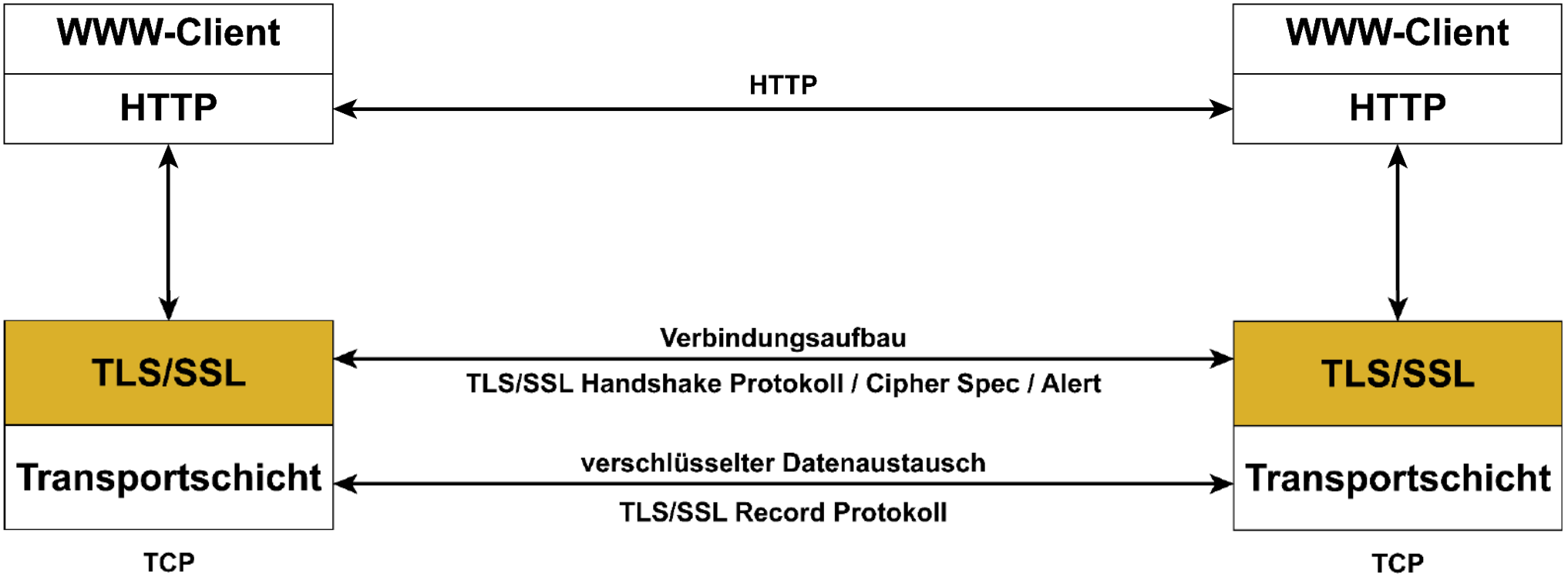
- ***Server-/Client-Random*** Zufallszahlen, die von Server und Client für jede Verbindung neu erzeugt werden.
- ***Server-Write-MAC-Secret*** Geheimer Schlüssel für die HMAC-Operationen on Daten, die zum Server übertragen werden.
- ***Client-Write-MAC-Secret*** Geheimer Schlüssel für die HMAC-Operationen von Daten, die zum Client übertragen werden.
- ***Server-Write-Key*** Symmetrischer Schlüssel für die Verschlüsselung vom Server und für die Entschlüsselung vom Client.
- ***Client-Write-Key*** Symmetrischer Schlüssel für die Verschlüsselung vom Client und für die Entschlüsselung vom Server.
- ***Initialization-Vectors*** Wird genutzt, wenn ein entsprechender Mode of Operation mit Initialization Vector (IV)verwendet wird.
- ***Sequence-Numbers*** Sowohl Client als auch Server nutzen Sequenznummern für die gesendeten und empfangenen Daten jeder Verbindung. Falls der Client oder der Server ein Change Cipher Spec verschickt oder erhält, wird die Sequenznummer wieder auf Zero gesetzt.

  Sequenznummern können nicht größer als 2^{64} -1 werden.

Sitzungs- und Verbindungskonzepte
→ Session/Connection

- An den Zuständen der Verbindungen wird deutlich, dass die verwendeten Schlüssel jeweils nur für eine Verbindung gelten.
- Für alle Verbindungen einer Sitzung werden die gleichen Verfahren genutzt.
- D.H. bei der erneuten Verwendung einer bereits bestehenden Verbindung kann auf das Aushandeln der Verfahren im Handshake verzichtet werden.

Übersicht des Protokollablaufs
→ TLS/SSL



Protokollablauf
→ Schritte und Phasen

- Der Protokollablauf bei TLS/SSL erfolgt in **zwei Schritten**.

- **1. Schritt: Verbindungsaufbau**, unterteilt in **vier Phasen**:

  - *1. Phase:* Aushandlung der Sicherheitsparameter.

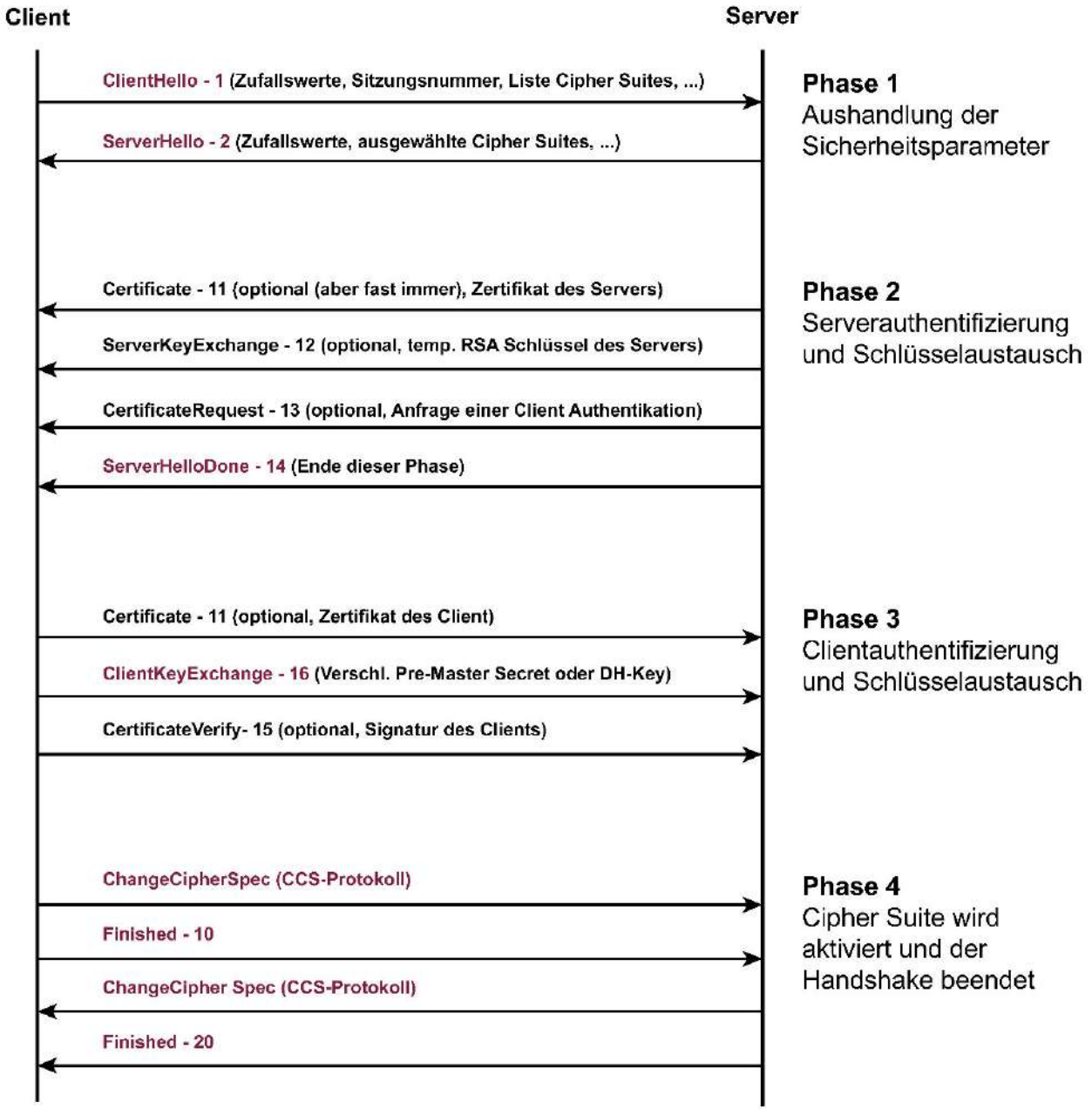
  - *2. Phase:* Serverauthentisierung (Optional) und Schlüsselaustausch

  - 3. *Phase:* Clientauthentisierung **(Optional)** und Schlüsselaustausch

  - *4. Phase:* Beendigung des Handshakes

- **2. Schritt: Transfer-Mode**
  verschlüsselte und *integritätsgesicherte* Datenübertragung

TLS/SSL
→ Verbindungsaufbau

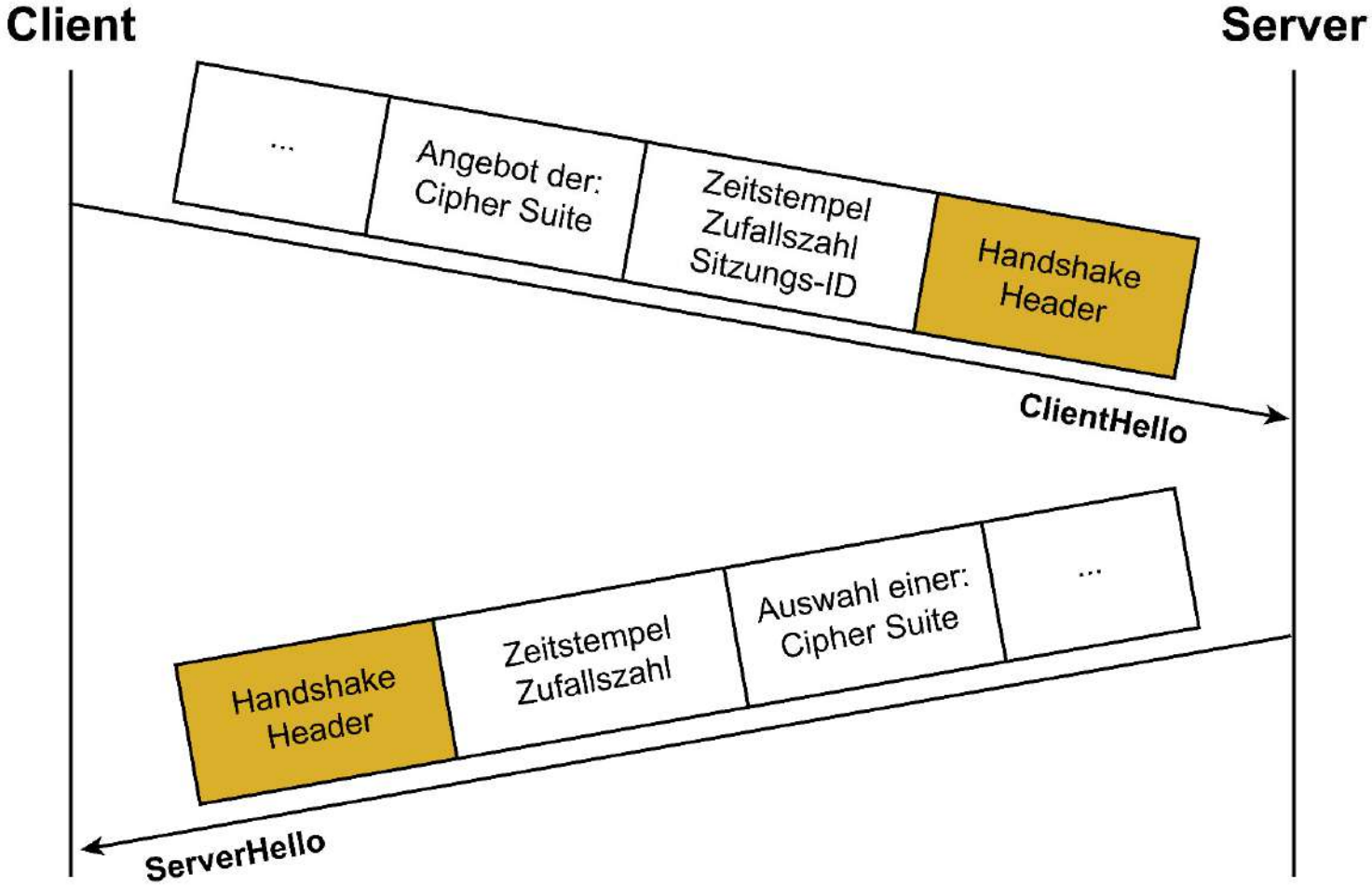


TLS/SSL - Verbindungsaufbau
→ Phase 0: HelloRequest

- Der Server **kann** ein “HelloRequest (0)” immer zum Client senden.
- Diese Nachricht hat keine Parameter.
- Sie wird vom Server geschickt, um den Client zu einem „ClientHello“ zu veranlassen.
  - Client antwortet nur, wenn er sich nicht in einem Handshake befindet.
- Soll nicht zum Aufbau einer Verbindung genutzt werden(!), dies ist die Aufgabe des Clients mit ClientHello (nächste Folie).
- Erhält der Server auf ein HelloRequest keine Antwort, **kann** die Verbindung geschlossen werden.

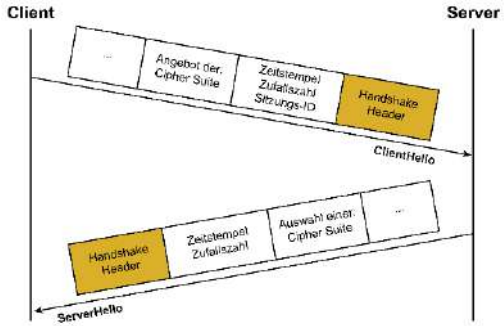
TLS/SSL - Verbindungsaufbau
→ Phase 1: Übersicht

Aushandlung der Sicherheitsparameter



TLS/SSL - Verbindungsaufbau
→ Phase 1: ClientHello

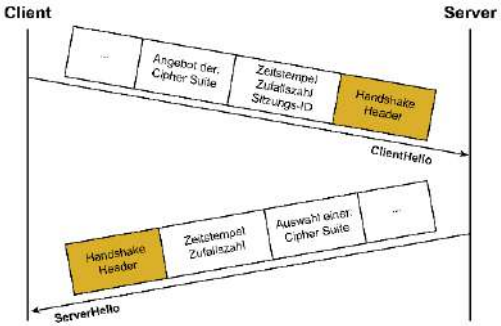
- In der ersten Phase werden die Sicherheitsparameter festgelegt.
- Mit dem Nachrichten-Typ „ClientHello (1)“ startet der Client den Aufbau der TLS/SSL-Verbindung.
- In einer bereits aktiven TLS/SSL-Verbindung führt diese Nachricht zu einer Neuverhandlung der Sicherheitsparameter.
- In dieser Klartextnachricht sind bereits wichtige Informationen zur Erzeugung des gemeinsamen geheimen Schlüssels enthalten:
  - Random Client - RC (4 Byte Zeitstempel + 28 Byte Zufallszahl)
  - Sitzungs-ID,
  - Prioritätenliste der Cipher Suites (Kryptographie- und Kompressionsverfahren), die der Client unterstützt.



- Der Client sollte **nur** kryptographische Verfahren und Funktionen sowie Schlüssellängen für die **Chiper Suite anbieten**, die sein gewünschtes **Cyber-Sicherheitsniveau erfüllen**.

TLS/SSL - Verbindungsaufbau
→ Phase 1: ServerHello

- Mit dem Nachrichten-Typ „ServerHello (2)“ antwortet der Server auf das „ClientHello“ des Clients.
- Die Nachricht enthält die gleichen Parameter wie der Nachrichten-Typ „ClientHello“, teilweise allerdings mit leicht abgewandelter Bedeutung.
  - Random Server - RS (4 Byte Zeitstempel + 28 Byte Zufallszahl)
- Zudem enthält dieser Nachrichten-Typ die ausgewählte Cipher Suite vom Server, die Client und Server nachfolgend nutzen.
- Hat der Client eine Sitzungs-ID vorgeschlagen, überprüft der Server diese Sitzungs-ID und übernimmt sie, sofern diese Sitzung noch nicht zu lange zurückliegt.
- Gibt der Server keine Sitzungs-ID an, so bedeutet dies, dass die gegenwärtige Sitzung später nicht erneut aufgenommen wird.



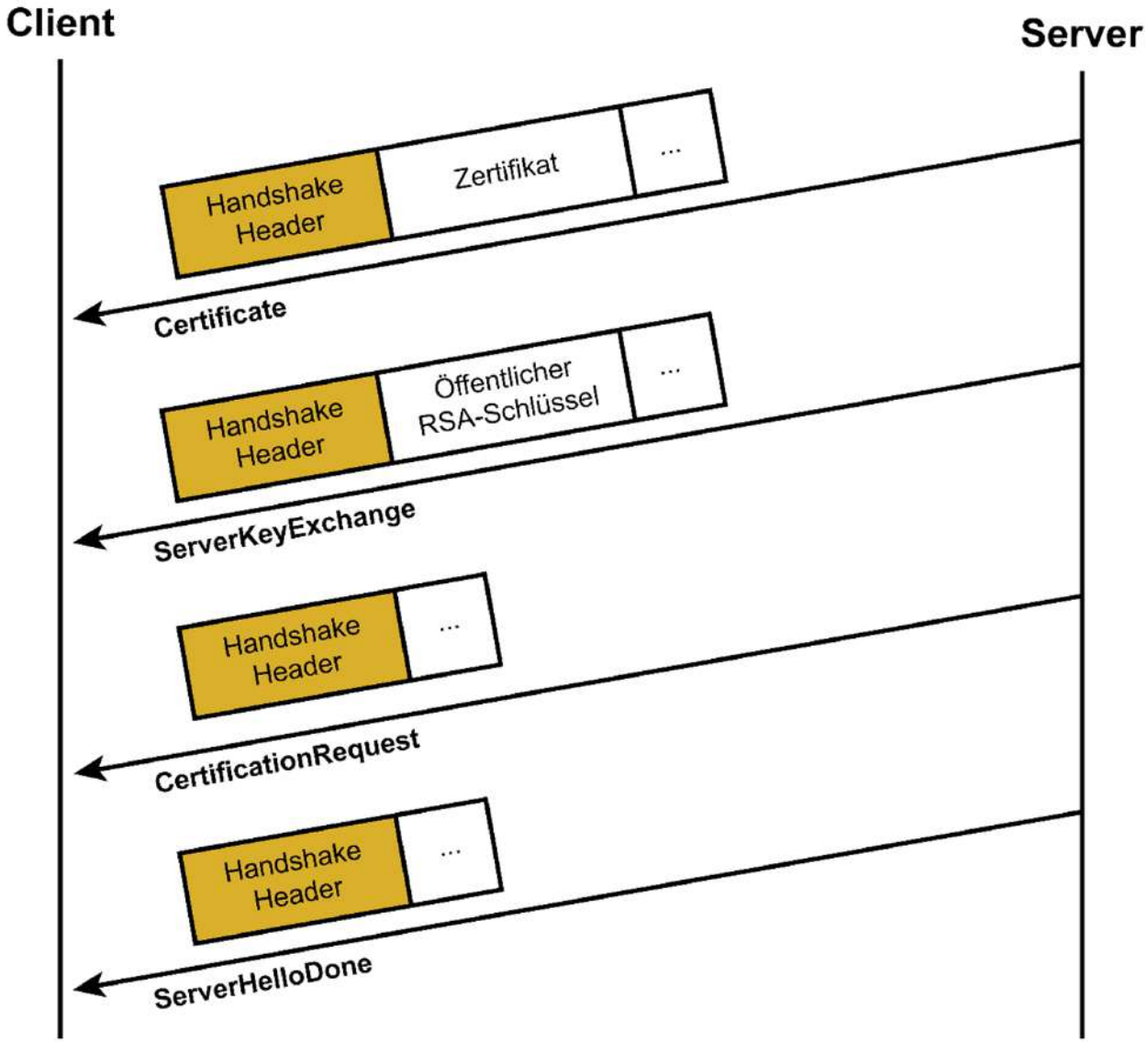
- Der Server sollte **nur kryptographische Verfahren** und **Funktionen** sowie Schlüssellängen auswählen, die sein gewünschtes **Cyber-Sicherheitsniveau erfüllen**.

TLS/SSL
→ Cipher Suites

- Beim **“Server/Client Hello”** werden die sogenannten **Cipher Suites** ausgewählt.
- Dabei handelt es sich um eine Kombination aus Schlüsselaustauschverfahren, Verschlüsselungsverfahren mit Schlüssellänge und ein Verfahren zum Integritätscheck.
- Jede Cipher Suite definiert mögliche Kombinationen aus Schlüsselaustauschverfahren (DHE_RSA), Verschlüsselungsalgorithmus (AES)/Schlüssellänge (256)/Mode of Operation (GCM) und die One-Way-Hashfunktion (SHA384) für den HMAC.

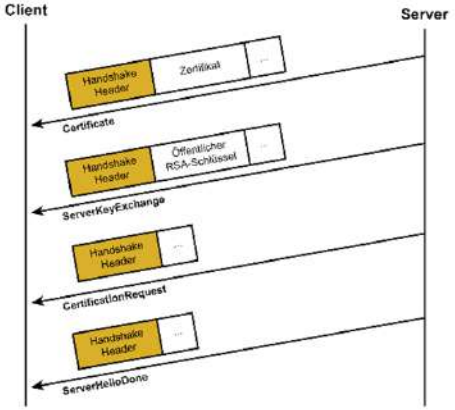
TLS/SSL - Verbindungsaufbau
→ Phase 2: Übersicht

Serverauthentifizierung und Schlüsselaustausch



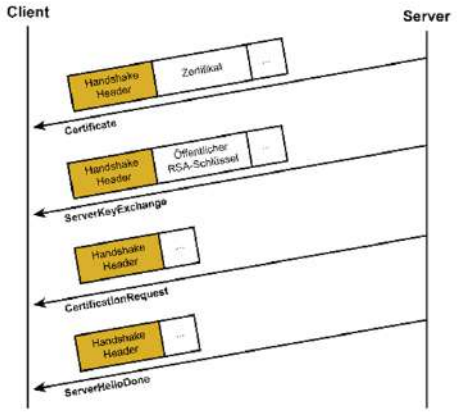
TLS/SSL - Verbindungsaufbau
→ Phase 2: Certificate

- In der zweiten Phase finden die implizierte **Serverauthentifizierung** und der **Schlüsselaustausch** statt.
- Soll der Server authentifiziert werden, so muss er dem Client im Nachrichten-Typ “Certificate (11)” sein Zertifikat zukommen lassen.
- Beim Server handelt es sich im Normalfall um ein X.509v3-Zertifikat, das für eine Domäne einer Organisation (zum Beispiel internet-sicherheit.de) von einer Zertifizierungsinstanz ausgegeben wurde.
- Dabei muss das Certificate zu der Cipher Suite passen!
- Wird RSA verwendet, muss das Zertifikat einen RSA-Schlüssel enthalten etc.



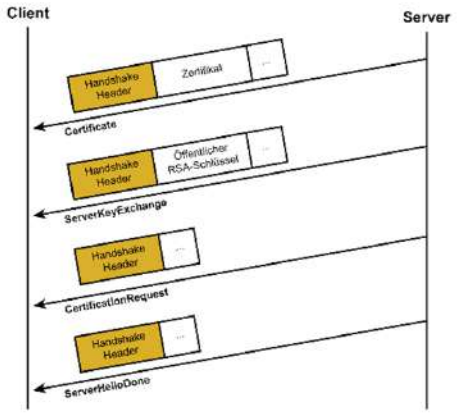
TLS/SSL - Verbindungsaufbau
→ Phase 2: ServerKeyExchange

- Diese Meldung wird nicht gesendet, wenn der Server ein Zertifikat mit festen Diffie-Hellman Parametern oder mit einem RSA-Schlüssel besitzt.
- Versendet der Server kein Zertifikat, so lässt er dem Client im Nachrichten-Typ “ServerKeyExchange (12)” einen temporären öffentlichen RSA-Schlüssel zukommen.



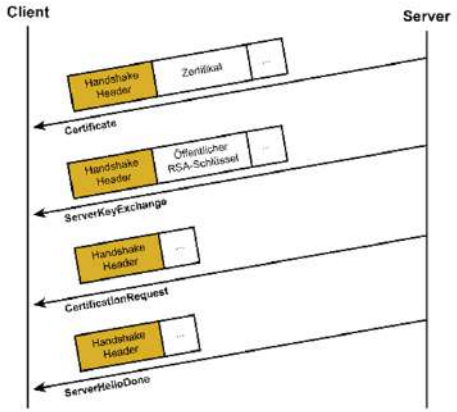
TLS/SSL - Verbindungsaufbau
→ Phase 2: CertificationRequest

- Falls der Server auch eine Authentifikation des Client fordert, sendet er den Nachrichten-Typ „CertificationRequest (13)“.



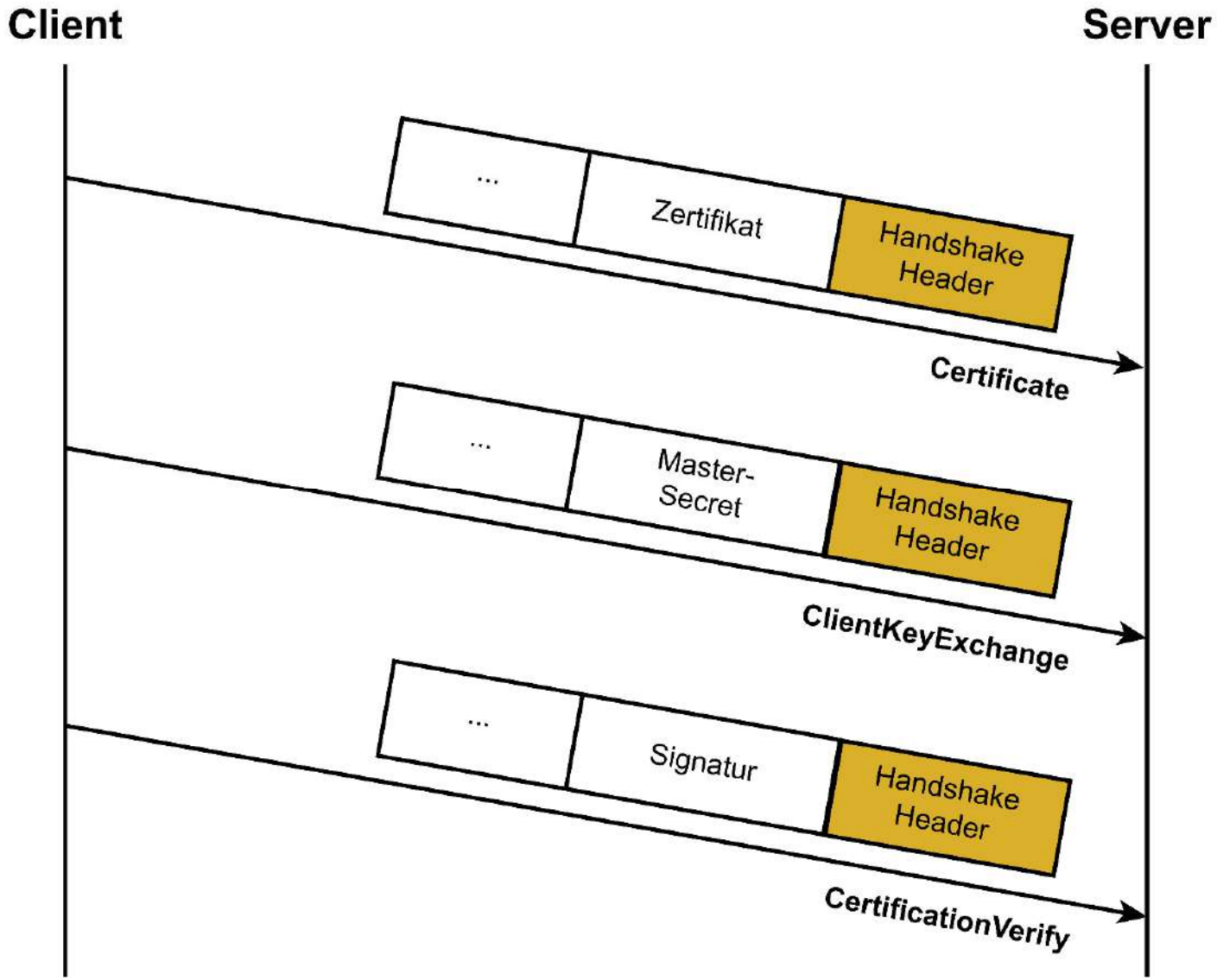
TLS/SSL - Verbindungsaufbau
→ Phase 2: ServerHelloDone

- Der Server beendet seine Übertragung dieser Phase mit dem Nachrichten-Typ „ServerHelloDone (14)“.
- Dieser Nachrichten-Typ ist notwendig, da die Nachrichten-Typen „Certificate“, „ServerKeyExchange“ und „CertificateRequest“ nicht notwendigerweise geschickt werden.
- So erkennt der Client das Ende der Phase 2.



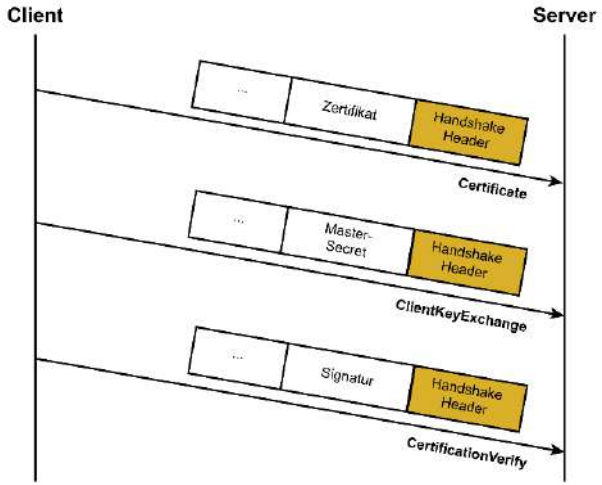
TLS/SSL - Verbindungsaufbau
→ Phase 3: Übersicht

Clientauthentifizierung und Schlüsselaustausch



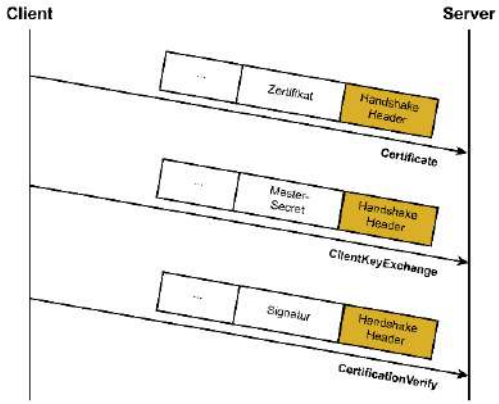
TLS/SSL - Verbindungsaufbau
→ Phase 3: Certificate

- In der Phase 3 kann der Client vom Server authentisiert werden und der Schlüsselaustausch wird fortgeführt.
- Soll der Client authentifiziert werden, so muss er dem Server sein Zertifikat im Nachrichten-Typ „Certificate (11)“ zukommen lassen.



TLS/SSL - Verbindungsaufbau
→ Phase 3: ClientKeyExchange

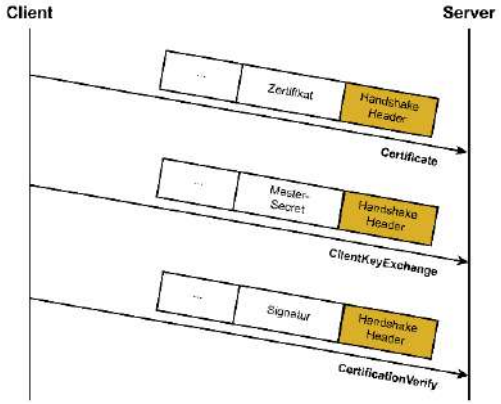
- Der Client prüft zunächst die Gültigkeit des Server-Zertifikats.
- Anschließend übermittelt er dem Server mit dem Nachrichten-Typ „ClientKeyExchange (16)“ seine geheime Basisinformation mithilfe des Kryptoverfahrens aus der ausgewählten Cipher Suite, das 48 Byte Pre-Master-Secret (Pre).
- Haben sich Client/Server auf das RSA-Verfahren geeinigt, so verschlüsselt der Client das Pre-Master-Secret mit dem öffentlichen Schlüssel des Servers aus dem „Certificate“ (Server-Zertifikat) oder aus dem „ClientKeyExchange“.
- Beim Diffie-Hellman-Schlüsselaustauschverfahren sendet der Client nur seinen öffentlichen Schlüssel an den Server zurück.



- Das Diffie-Hellman-Verfahren wird hier nicht zur Berechnung des geheimen Schlüssels genutzt, sondern zur dezentralen Berechnung des Pre-Master-Secrets (Pre).

TLS/SSL - Verbindungsaufbau
→ Phase 3: CertificateVerify

- Mit dem Nachricht-Typ „CertificateVerify (15)“ signiert der Client Informationen, sofern er sie in dieser Phase gesendet hat.
- Client „beweist“ dem Server, dass er im Besitz des geheimen Schlüssels für das Zertifikat ist.



- **Ablauf:**
  - Client berechnet einen Hashwert über die Bytes aller bisher ausgetauschten Handshake-Nachrichten, von ClientHello bis ClientKeyExchange.
  - Client signiert diesen Hashwert mit seinem geheimen Schlüssel.
  - Der Server berechnet einen Hashwert über die gleichen Handshake-Nachrichten und verifiziert die Signatur mit dem öffentlichen Schlüssel aus dem Zertifikat.

TLS/SSL - Verbindungsaufbau
→ Master Secret Berechnung

Das Pre-Master-Secret und die ausgetauschten Zufallszahlen werden nun von Client und Server genutzt, um das 48 Byte Master-Secret zu berechnen, aus dem die benötigten geheimen Schlüssel (Session Keys, Key zur HMAC-Berechnung, …) abgeleitet werden.

Berechnung des Master-Secrets :

Master-Secret = **KH (** Pre-Master-Secret, ClientHello.random || ServerHello.random **)**

KH = „Keyed-Hashing for Message Authentication-Verfahren; HMAC-Verfahren

TLS/SSL - Verbindungsaufbau
→ Schlüsselerzeugung

- Alle Schlüssel (Session Keys, Key zur HMAC-Berechnung, ...) werden nach dem einen gleichartigen Schema sowohl vom Client als auch vom Server aus dem Master-Secret abgeleitet.
- Dazu generiert das Protokoll so lange eine Folge von Schlüsselblöcken „Key-Block“, bis alle Schlüssel damit konstruiert sind.

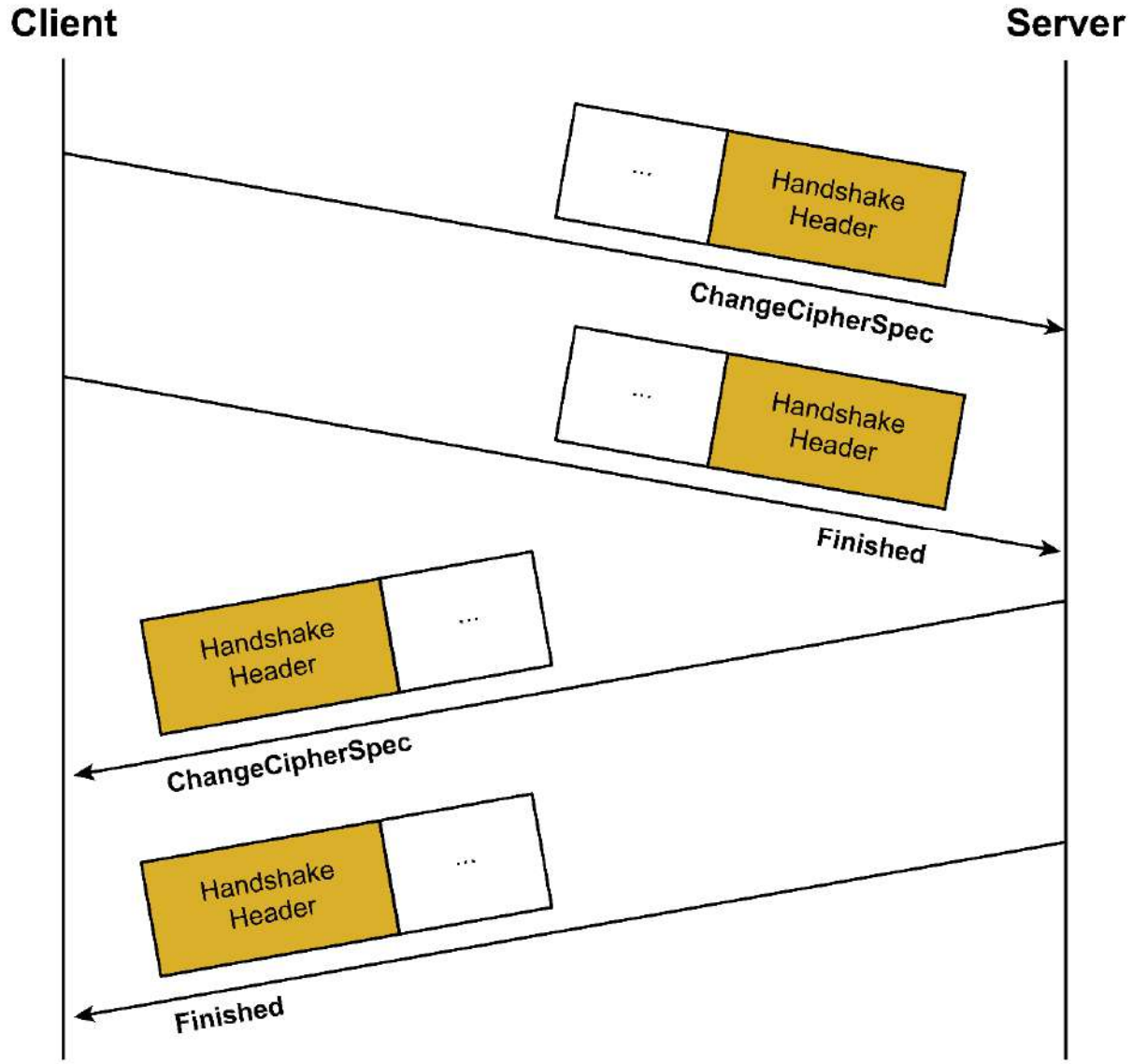
Berechnung des Key-Blocks (Schlüsselblock)

Key-Block = **KH (** Master-Secret, ServerHello.random || ClientHello.random **)**

KH = „Keyed-Hashing for Message Authentication“-Verfahren; HMAC-Verfahren

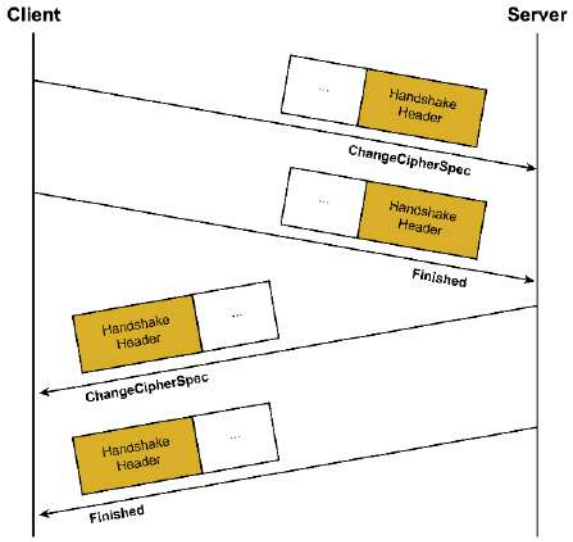
TLS/SSL - Verbindungsaufbau
→ Phase 4: Übersicht

Cipher Suite wird aktiviert und der Handshake beendet



TLS/SSL - Verbindungsaufbau
→ Phase 4: ChangeCipherSec + Finished

- Die Phase 4 beendet den Handshake.
- Mit der „ChangeCipherSpec“-Nachricht (CCS-Protokoll) zeigen Client und Server, dass sie ab jetzt die ausgehandelten Verfahren/Parameter nutzen.
- Mit der „Finished (20)“-Nachricht symbolisieren beide, dass ihr Teil des Verbindungsaufbaus abgeschlossen ist.
- Diese Meldung ist die erste Nachricht, die bereits mit den neuen Sicherheitsparametern für das Record Protokoll behandelt wird.



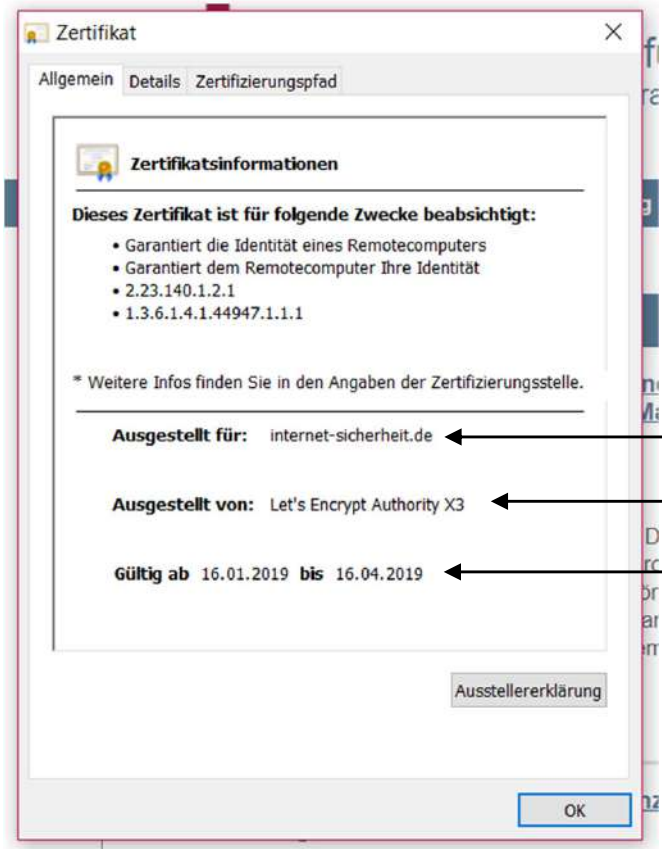
TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- **Domänen-Zertifikaten**
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

TLS/SSL
→ Domänen-Zertifikate

- **Domänen-Zertifikate helfen, Attribute von Domänen zu überprüfen.**
- Wesentliche **Inhalte eines Domänen-Zertifikates** sind:
  - Name der Organisation, deren Authentizität durch dieses Domänen-Zertifikat bestätigt wird
  - Public-Key der Organisation (Domäne)
  - Name der ausstellenden Zertifizierungsstelle
  - Gültigkeit des Domänen-Zertifikats
- **Aufgabe eines Domänen-Zertifikats:**
  - Eindeutige Zuordnung eines Public-Key's zu einer Organisation.
  - Für die Korrektheit dieser Zuordnung und dass die Organisation, die den passenden geheimen Schlüssel besitzt, auch existiert, verbürgt sich die ausstellende Zertifizierungsstelle.
  - Diese signiert das Domänen-Zertifikat, wodurch es für Dritte ohne die Kenntnis des geheimen Schlüssels der Zertifizierungsstelle unmöglich wird, das Domänen-Zertifikat zu verändern.
- **Der Aufbau von Zertifikaten ist standardisiert (RFC 2459):**

TLS/SSL – Domänen-Zertifikate
→ Zertifikatsinformationen – if(is)

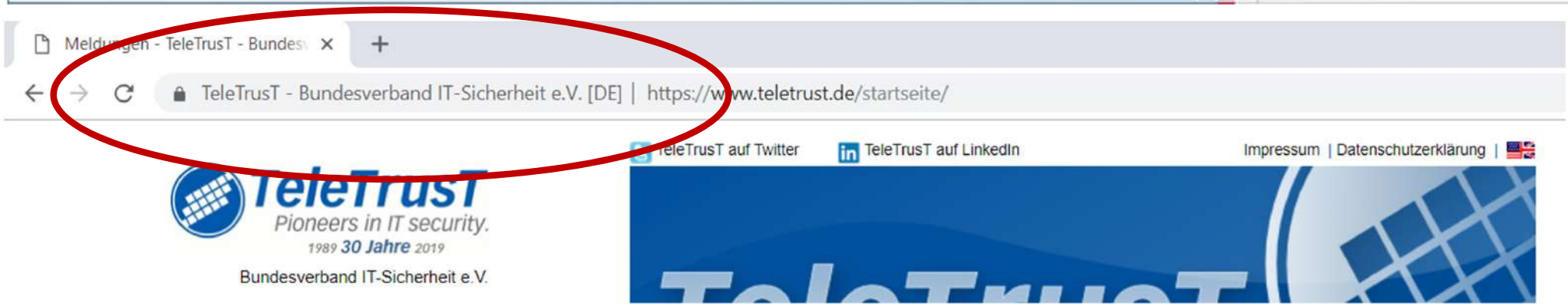


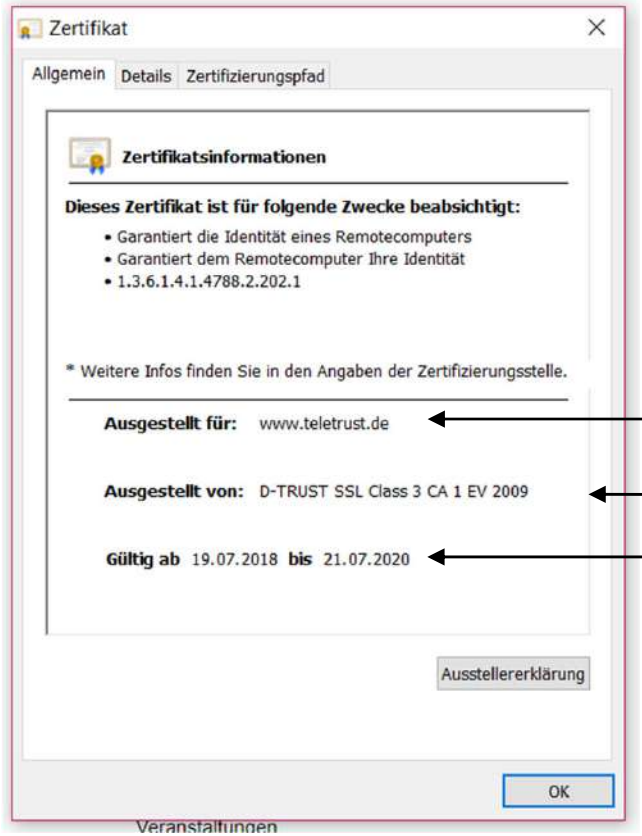
Domänen-Namen

Herausgeber des Zertifikates

Gültigkeitszeitraum

TLS/SSL – Domänen-Zertifikate
→ Zertifikatsinformationen – if(is)





Domänen-Namen

Herausgeber des Zertifikates

Gültigkeitszeitraum

TLS/SSL – Domänen-Zertifikate
→ Wichtige Aspekte (1/2)

- Eine Domäne muss schon **registriert sein, bevor** ein TLS/SSL-Zertifikat **beantragt wird**, weil Zertifizierungsstellen, die ein Zertifikat ausstellen, die Domain-Inhaberschaft überprüfen müssen.

- Bezüglich der Überprüfung der Domain-Inhaberschaft gibt es verschiedene Vertrauensstufen:
  - Domain Validated (DV)
  - Organisation Validated (OV)
  - Extended Validation (EV)

- **Domain Validated (DV)-Zertificate:**
  - DV-Zertifikate sind die **einfachste Art** von TLS/SSL-Zertifikaten.
  - DV steht für Domain Validation (Validierung/Überprüfung der Domain).
  - Eine Zertifizierungsstelle bestätigt damit, dass zum Beispiel der Website-Inhaber die **administrative Kontrolle über die Domain** nachweisen kann.
  - Sie enthalten wenige Identitätsinformationen im Zertifikat, zum Beispiel enthalten sie keinerlei Unternehmensinformationen.

TLS/SSL – Domänen-Zertifikate
→ Wichtige Aspekte (2/2)

- **Organization Validated (OV)-Zertificate:**
  - OV-Zertifikate enthalten eine Unternehmensverifizierung.
  - Das heißt, es sind Informationen zum jeweiligen Unternehmen enthalten.
  - Im Gegensatz zu EV-Zertifikaten werden diese Informationen allerdings nicht so prominent angezeigt.
  - Um die Identitätsdaten eines Unternehmens zu erkennen, müssen sich die Besucher der Webseite die Zertifikatdetails ansehen.

- **Extended Validation (EV)-Zertificate:**
  - EV-Zertifikate enthalten die meisten Unternehmensdaten, und Firmen müssen bei dieser Variante die strengsten Anforderungen erfüllen, bevor sie ein solches TLS/SSL-Zertifikat erhalten.
  - Hier steht die verifizierte Identität eines Unternehmens im Mittelpunkt und verleiht so der Webseite **das höchste Maß an Glaubwürdigkeit**: der Name des betreffenden Unternehmens wird deutlich in der grünen Adresszeile eines Browsers angezeigt.

TLS/SSL – Domänen-Zertifikate
→ Root-Zertifikat

- Ein Root-Zertifikat ist das oberste Zertifikat im Verzeichnisbaum.
- Damit ein Domain-Zertifikat vom Browser geprüft werden kann, muss dem Browser das verwendete Root-Zertifikat der Zertifizierungsstelle bekannt sein.
- Dazu können Root-Zertifikate zusätzlich zu den im Browser schon vorhandenen installiert werden.
- Die gängigsten Browser haben heutzutage schon die wichtigsten Root-Zertifikate der größten Zertifizierungsstellen vorinstalliert.
- Einmal akzeptierte Zertifikate können natürlich auch im Browser verwaltet, also aufgelistet und gegebenenfalls gelöscht werden.
- Besucht man eine Seite, deren Zertifikat bzw. Zertifizierungsstelle dem Browser unbekannt ist, so erhält der Nutzer einen Hinweis, mit der Möglichkeit, das Zertifikat zu prüfen, ihm einmalig oder dauerhaft zu vertrauen, oder das Zertifikat in den Browser zu laden, wodurch zukünftig an dieser Stelle kein Hinweis mehr erscheinen würde.

TLS/SSL -Zertifikate
→ TLS/SSL Authentikation → Bewertung

- **Die Qualität der TLS/SSL-Client-Authentikation hängt ab von:**
  - Der Vertrauenswürdigkeit der Zertifizierungsinstanz
  - Dem Level, auf dem die Zertifizierungsinstanz die Authentizität des Teilnehmers überprüft
  - Der Bereitstellung einer CRL (Certificate Revocation List) durch die Zertifizierungsinstanz

- **Verschiedene Zertifizierungs-Anbieter bieten unterschiedliche Klassen von Zertifikaten an:**
  - Ein Zertifikat der Klasse 1 bekommt der Teilnehmer schon nach der Zusendung des Namens und der E-Mail-Adresse, ohne dass seine Identität näher geprüft wird.
  - Für ein Zertifikat der Klasse 2 wird z.B. die Zusendung einer Kopie des Ausweises oder des Führerscheins verlangt.
  - Bei höheren Klassen muss der Teilnehmer sich persönlich (z.B. mit Hilfe des Ausweises) bei einer Registration Authority (RA) authentisieren.
  - Hierbei stellt sich die Frage, wie vertrauenswürdig eine solche Registration Authority (RA) ist.

TLS/SSL-Technologie
→ Inhalt

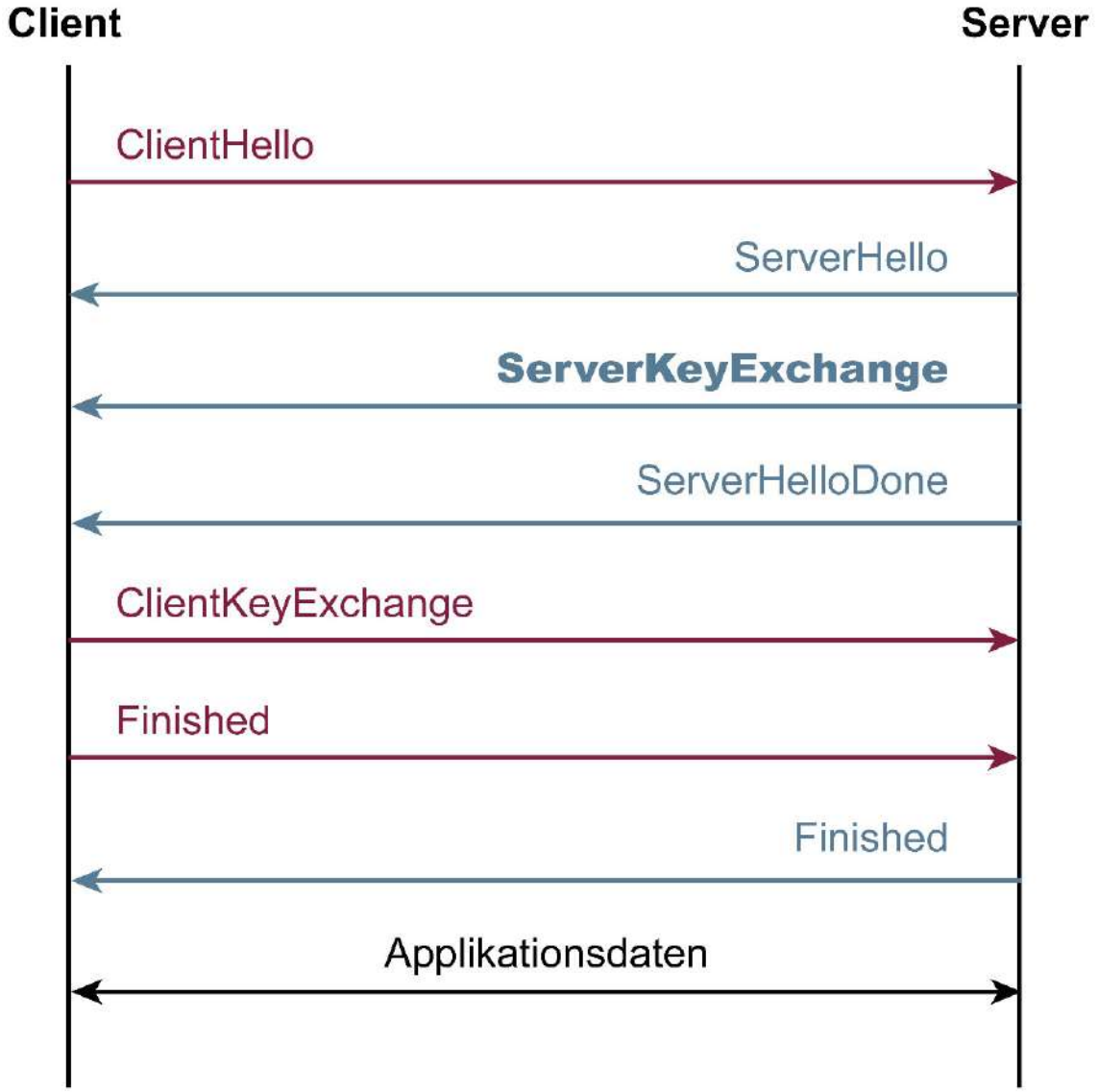
- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- **TLS/SSL Authentifikationsmethoden**
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Authentifikationsmethoden
→ Einführung

- TLS/SSL benutzt Public Key Zertifikate für die Authentifizierung von Server und Client.
- **Es werden drei 3 Verbindungsarten unterschieden:**
  - 1. Server und Client ohne Authentifizierung
  - 2. Server authentifiziert, Client anonym (gängigste Art)
  - 3. Server und Client authentifiziert

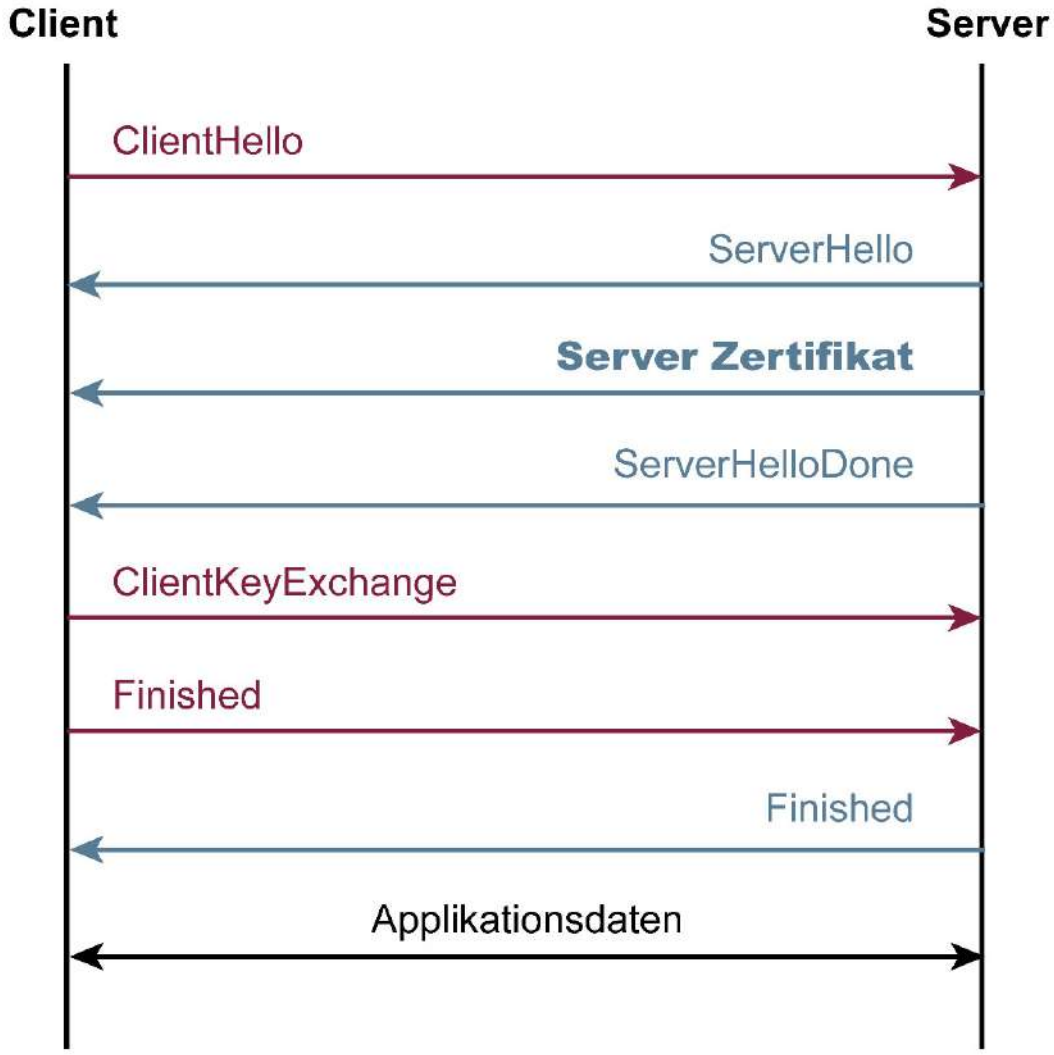
Authentifikationsmethoden
→ Server und Client ohne Authentifizierung

- Bei dieser Verbindungsart verlangt weder der Server noch der Client eine Authentifizierung seines Kommunikationspartners durch ein Zertifikat.
- Nach Beendigung des Verbindungsaufbaus können Applikationsdaten ausgetauscht werden.
- Die **beiderseits anonyme Verbindung** ist nicht sicher gegenüber aktiven Man-In-The-Middle-Angriffen.
- **Diese Verbindungsart sollte daher vermieden werden!**



Authentifikationsmethoden
→ Server authentifiziert, Client anonym

- Hier teilt der **Server** seinen öffentlichen Schlüssel dem Client nicht durch eine ServerKeyExchange-Nachricht mit, sondern durch die **Übermittlung seines Zertifikates**.
- Kann der Client das **Zertifikat verifizieren**, so kann sich dieser **sicher sein**, dass nach einem erfolgreichen Verbindungsaufbau eine TLS/SSL-Verbindung zu genau jenem Server zustande gekommen ist, dessen Zertifikat empfangen wurde.
- Hier kann sich ein Man-In-The-Middle in Richtung des Clients nun nicht mehr als falscher Server ausgeben, womit durch die Server-Authentifizierung aktive Angriffe verhindert werden.



Authentifikationsmethoden
→ Server Authentikation im Detail

Ablauf:

1. Client wählt Internetseite an

2. Server übermittelt dem Client sein öffentliches Domänen-Zertifikat mit dem öffentlichen RSA-Schlüssel

3. Client prüft das Domänen-Zertifikat auf Gültigkeit

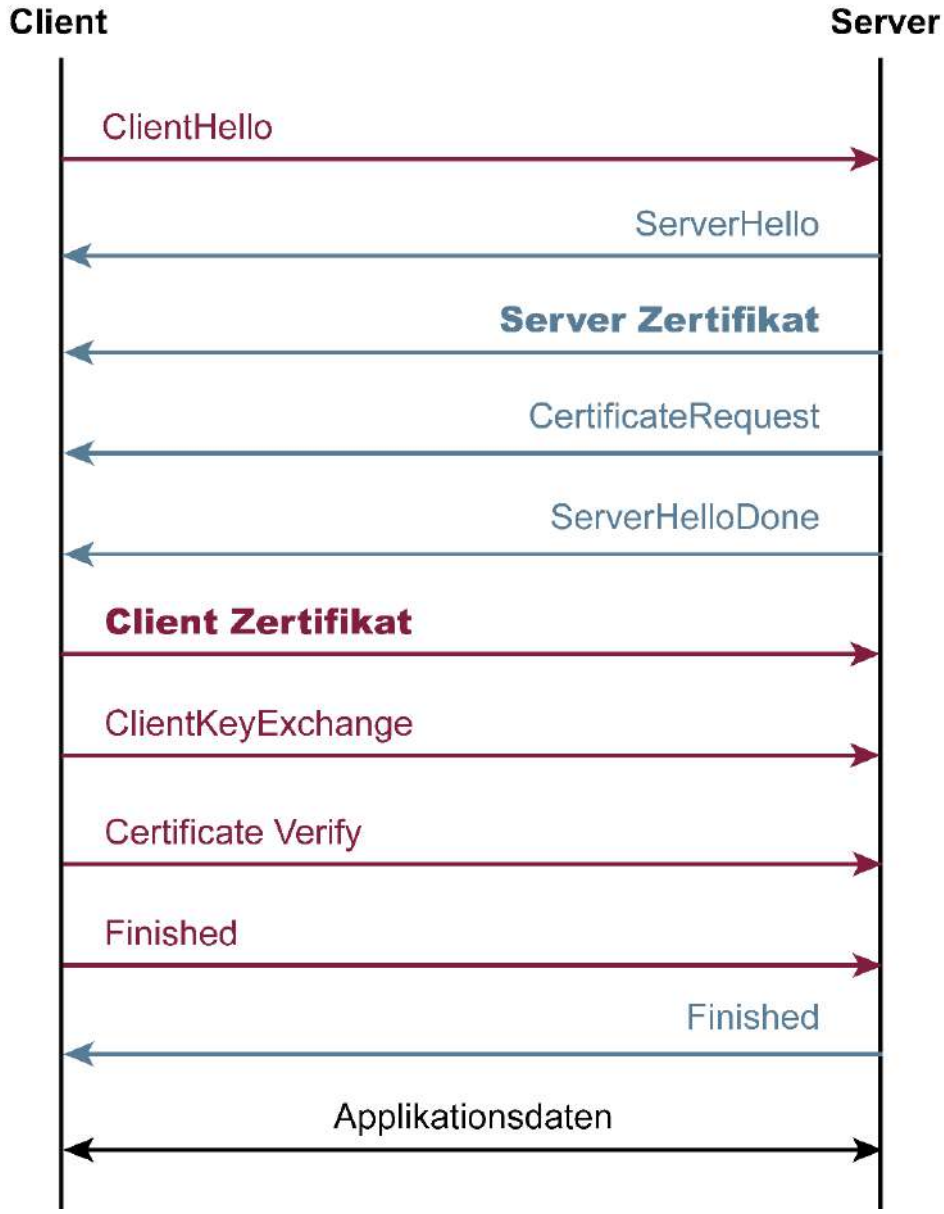
4. Client generiert Zufallszahl (Pre-Master Secret), verschlüsselt diese für den Server mit dem öffentlichen RSA aus dem Domänen-Zertifikat

5. Server entschlüsselt die Zufallszahl (Pre-Master Secret), generiert die Session Keys und verschlüsselt die Daten nun mit dem Sessionkey für den Client.

6. Verschlüsselter Datenaustausch zwischen Server und Client, und damit die Gewissheit, dass der Server der echte ist.

Authentifikationsmethoden
→ Server und Client authentifiziert

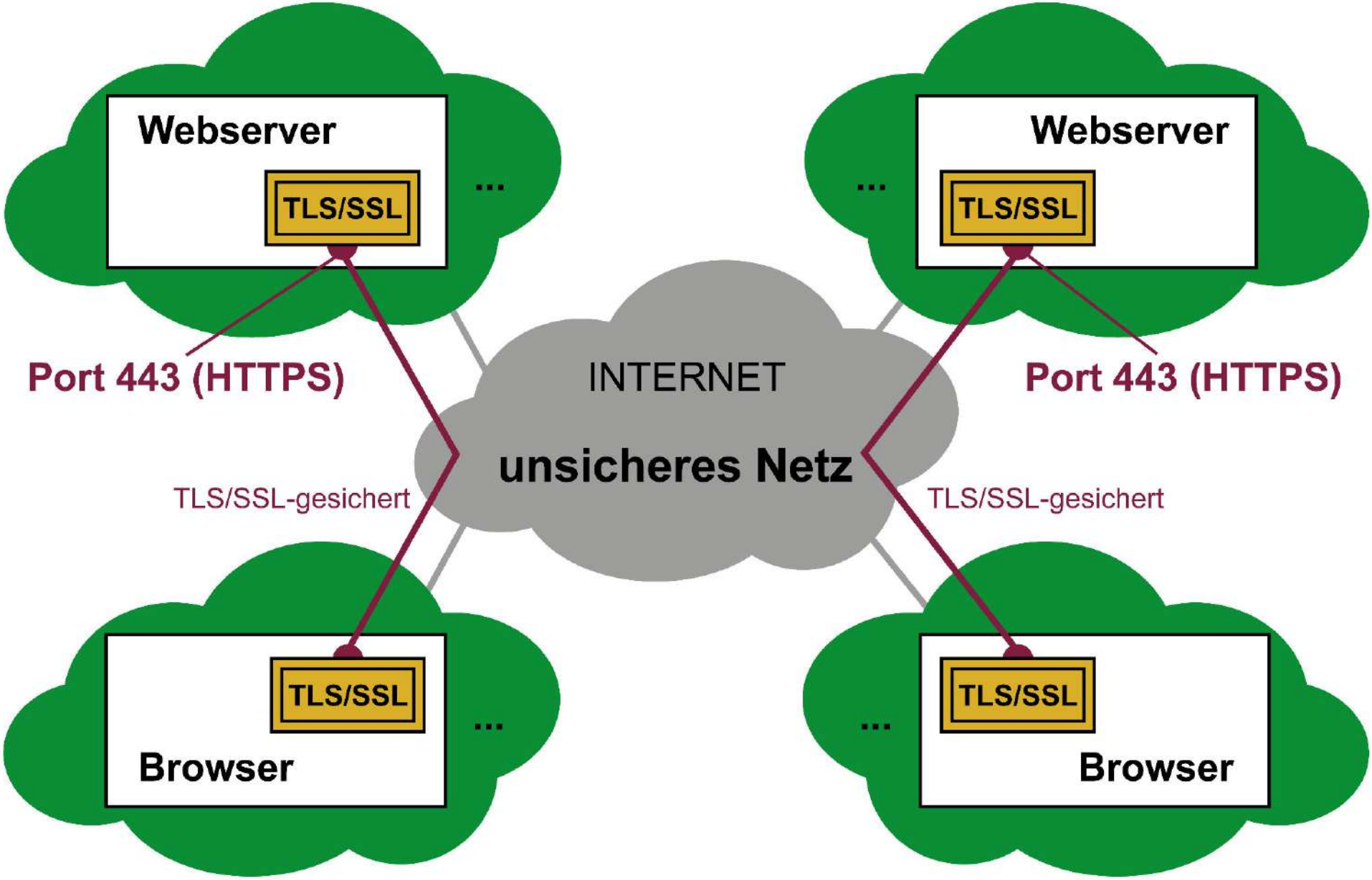
- Der Server kann mit der CertificateRequest-Nachricht auch eine Client-Authentifikation über ein Zertifikat des Clients beantragen.
- Dieser stellt das Zertifikat mit der ClientCertificate-Nachricht zu und beweist mit der CertificateVerify-Nachricht, dass er auch tatsächlich jener ist, dessen Name im Zertifikat erscheint.
- Falls der Client kein Zertifikat besitzt, welches in der vom Server bekannt gegebenen Liste auftaucht, wird der Verbindungsaufbau abgebrochen.



TLS/SSL-Technologie
→ Inhalt

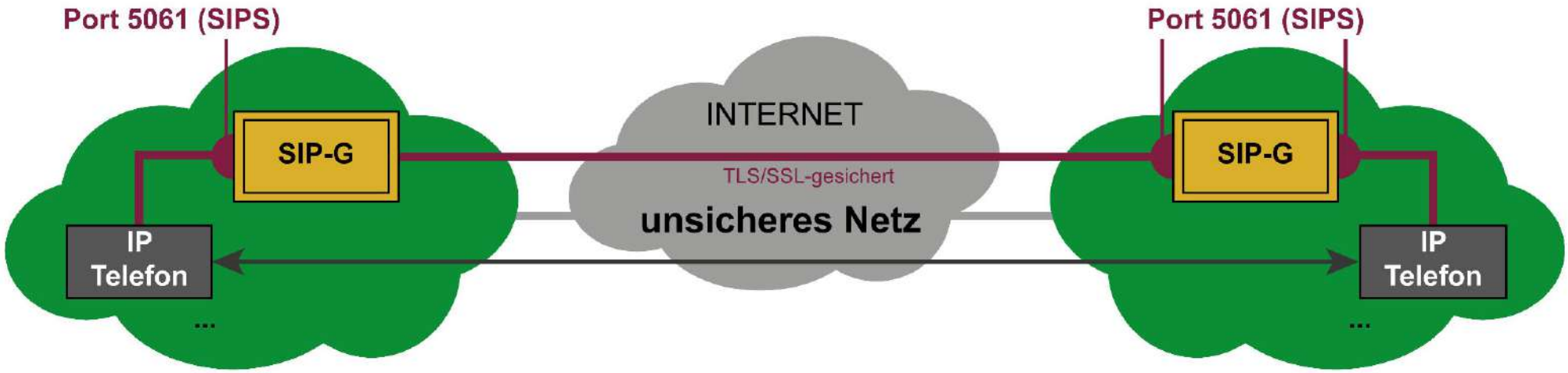
- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- **TLS/SSL Anwendungsformen**
- TLS/SSL Protokollmitschnitt
- Zusammenfassung

Anwendungsformen
→ Gesicherte Web-Kommunikation



Anwendungsformen
→ Gesicherte Mail-Kommunikation

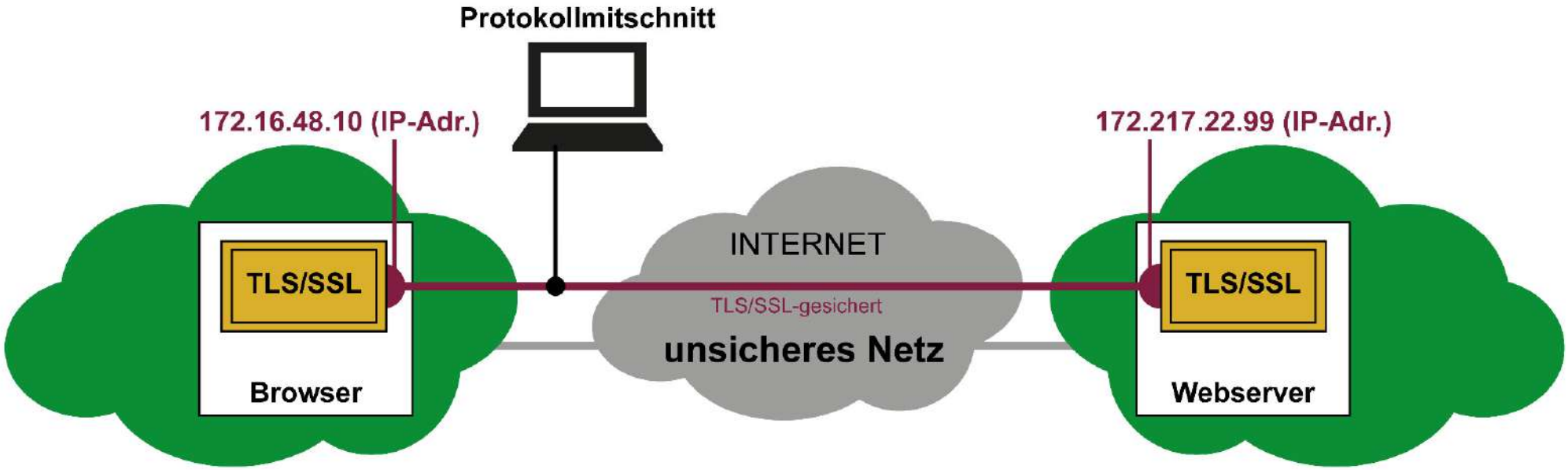
Anwendungsformen
→ Gesicherte SIP-Kommunikation



TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- **TLS/SSL Protokollmitschnitt**
- Zusammenfassung

Protokollmitschnitt
→ TLS/SSL-Protokoll - Übersicht



Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (1/11)

TCP-Verbindungsaufbau

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 0, Len: 0
Flags: 0x002 (SYN)

Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0 **C <<< S**
Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)
Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 0, Ack: 1, Len: 0
Flags: 0x012 (SYN, ACK)

Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 1, Ack: 1, Len: 0
Flags: 0x010 (ACK)

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (2/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

Frame 4: 236 bytes on wire (1888 bits), 236 bytes captured (1888 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 1, Ack: 1, Len: 170
Secure Sockets Layer
TLSv1.2 Record Layer: Handshake Protocol: Client Hello
Handshake Protocol: Client Hello
Random
gmt_unix_time: Jan 23, 2025 15:01:26.000000000 CET
random_bytes: 3f823c9ec03c535a245b6a5e78212356556bcf188d4b3a9f...
Cipher Suites (11 suites)
Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
Compression Methods (1 method)
Compression Method: null (0)
…

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (3/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

TCP- Bestätigung für Frame 4

Frame 5: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0 **C <<< S**

Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)

Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 1, Ack: 171, Len: 0

Flags: 0x010 (ACK)

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (4/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

Frame 6: 2780 bytes on wire (22240 bits), 2780 bytes captured (22240 bits) on interface 0 **C <<< S**
Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)
Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 1, Ack: 171, Len: 2714
    Flags: 0x018 (PSH, ACK)
Secure Sockets Layer
    TLSv1.2 Record Layer: Handshake Protocol: Server Hello
        Handshake Protocol: Server Hello
            Random
                gmt_unix_time: Aug 28, 2018 11:12:01.000000000 CEST
                random_bytes: 32b5bbb6291f31b56d071ceaee14e1e7be0863ac6c7c3a06...
            Session ID Length: 0
            Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
            Compression Method: null (0)
            Extensions Length: 24
            Extension: renegotiation_info
                Type: renegotiation_info (0xff01)
                Length: 1
                Renegotiation Info extension
                    Renegotiation info extension length: 0
                …

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (5/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

… *(Fame 6)*

TLSv1.2 Record Layer: Handshake Protocol: Certificate
Content Type: Handshake (22)
Version: TLS 1.2 (0x0303)
Length: 2289
Handshake Protocol: Certificate
Certificates Length: 2282
Certificates (2282 bytes)
Certificate Length: 1156
Certificate (id-at-commonName=www.google.de,id-at-organizationName=Google LLC,id-at-localityName=Mountain View,id-at-stateOrProvinceName=California,id-at-countryName=US)
signedCertificate
version: v3 (2)
serialNumber: 25340433
signature (sha256WithRSAEncryption)
Algorithm Id: 1.2.840.113549.1.1.11 (sha256WithRSAEncryption)
issuer: rdnSequence (0)
rdnSequence: 3 items (id-at-commonName=Google Internet Authority G3,id-at-organizationName=Google Trust Services,id-at-countryName=US)
…

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (6/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

… *(Fame 6)*

TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
- Content Type: Handshake (22)
- Version: TLS 1.2 (0x0303)
- Length: 333
- Handshake Protocol: Server Key Exchange
  - Length: 329
  - EC Diffie-Hellman Server Params
    - curve_type: named_curve (0x03)
    - named_curve: secp256r1 (0x0017)
    - Pubkey Length: 65
    - **pubkey: 04fa540dea1e37e332080d8d9ecafec2764a50ecb689b78d...**
    - Signature Hash Algorithm: 0x0401
      - Signature Hash Algorithm Hash: SHA256 (4)
      - Signature Hash Algorithm Signature: RSA (1)
    - Signature Length: 256
    - signature: 14ed6843402163b44f1d5699c329732ad3856c43b4792361...

TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done
- Content Type: Handshake (22)
- Version: TLS 1.2 (0x0303)
- Length: 4
- Handshake Protocol: Server Hello Done

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (7/11)

Aushandlung der Sicherheitsparameter, Serverauthentifizierung und Schlüsselaustausch

TCP- Bestätigung für Frame 6

Frame 7: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0 **C >>> S**

Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)

Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 171, Ack: 2715,Len0

Flags: 0x010 (ACK)

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (8/11)

Schlüsselaustausch, CCS, …

Frame 8: 192 bytes on wire (1536 bits), 192 bytes captured (1536 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 171, Ack: 2715, Len: 126
Flags: 0x018 (PSH, ACK)
Secure Sockets Layer
TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange
Handshake Protocol: Client Key Exchange
Length: 66
EC Diffie-Hellman Client Params
Pubkey Length: 65
pubkey: 0418395fd6c8855b0e43de39ad413466f3a4c513e8e58c2e...
TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
Content Type: Change Cipher Spec (20)
Version: TLS 1.2 (0x0303)
Length: 1
Change Cipher Spec Message
TLSv1.2 Record Layer: Handshake Protocol: Multiple Handshake Messages
Content Type: Handshake (22)
Version: TLS 1.2 (0x0303)
Length: 40
Handshake Protocol: Hello Request
Handshake Type: Hello Request (0)
Handshake Protocol: Hello Request

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (9/11)

CCS, …

Frame 9: 345 bytes on wire (2760 bits), 345 bytes captured (2760 bits) on interface 0 **C <<< S**
Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)
Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 2715, Ack: 297, Len: 279
    Flags: 0x018 (PSH, ACK)
Secure Sockets Layer
    TLSv1.2 Record Layer: Handshake Protocol: New Session Ticket
        Handshake Protocol: New Session Ticket
            Handshake Type: New Session Ticket (4)
            Length: 219
            TLS Session Ticket
                Session Ticket Lifetime Hint: 100800
                Session Ticket Length: 213
                Session Ticket: 00f7fe033d2ec55ea45919d7b87195bf3247ae08e9462ea4...
    TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
        Content Type: Change Cipher Spec (20)
        Version: TLS 1.2 (0x0303)
        Length: 1
        Change Cipher Spec Message
    TLSv1.2 Record Layer: Handshake Protocol: Multiple Handshake Messages
        Content Type: Handshake (22)
        Version: TLS 1.2 (0x0303)
        Length: 40
        Handshake Protocol: Hello Request

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (10/11)

Verschlüsselte und integritätsgesicherte Datenübertragung

Frame 10: 229 bytes on wire (1832 bits), 229 bytes captured (1832 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 297, Ack: 2994, Len: 163
Secure Sockets Layer
TLSv1.2 Record Layer: Application Data Protocol: http
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 158
Encrypted Application Data: 00000000000000019758b52e6af38f292ada9eafc1118770...

Frame 11: 454 bytes on wire (3632 bits), 454 bytes captured (3632 bits) on interface 0 **C >>> S**
Internet Protocol Version 4, Src: 172.16.48.10 (172.16.48.10), Dst: 172.217.22.99 (172.217.22.99)
Transmission Control Protocol, Src Port: 36102 (36102), Dst Port: https (443), Seq: 460, Ack: 2994, Len: 388
Secure Sockets Layer
TLSv1.2 Record Layer: Application Data Protocol: http
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 383
Encrypted Application Data: 0000000000000025d9a6a016536db15bbcfa2b4e37eb1a3...

...

Protokollmitschnitt
→ TLS/SSL-Protokoll – Teil (11/11))

Verschlüsselte und integritätsgesicherte Datenübertragung

Frame 12: 135 bytes on wire (1080 bits), 135 bytes captured (1080 bits) on interface 0 **C <<< S**
Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)
Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 2994, Ack: 460, Len: 69
Secure Sockets Layer
TLSv1.2 Record Layer: Application Data Protocol: http
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 64
Encrypted Application Data: 0000000000000000131f990c1519fe80b83ab43a6ec15eb65...

Frame 13: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface 0 **C <<< S**
Internet Protocol Version 4, Src: 172.217.22.99 (172.217.22.99), Dst: 172.16.48.10 (172.16.48.10)
Transmission Control Protocol, Src Port: https (443), Dst Port: 36102 (36102), Seq: 3063, Ack: 460, Len: 38
Secure Sockets Layer
TLSv1.2 Record Layer: Application Data Protocol: http
Content Type: Application Data (23)
Version: TLS 1.2 (0x0303)
Length: 33
Encrypted Application Data: 0000000000000000211199f018207c918fd98d8e0a3e7a5ca...

...

TLS/SSL-Technologie
→ Inhalt

- Ziele und Ergebnisse der Vorlesung
- Einleitung
- Architektur und Protokolle
- Protokollablauf: Prinzipien, Schritte und Phasen
- Domänen-Zertifikaten
- TLS/SSL Authentifikationsmethoden
- TLS/SSL Anwendungsformen
- TLS/SSL Protokollmitschnitt
- **Zusammenfassung**

TLS/SSL-Technologie
→ Zusammenfassung (1/3)

- **TLS (Transport Layer Security) / SSL (Secure Socket Layer)** ist ein Cyber-Sicherheitsprotokoll.
- TLS/SSL ist **applikationsunabhänig** und setzt logisch auf einem Transportprotokoll auf.
- TLS/SSL bündelt 3 Sicherheitsdienste:
  - **Authentifizierung** von Server und Client
  - **Verschlüsselung** der Daten
  - **Integritäts- und Autentizitätsüberprüfung** der Daten
- Gängigste Protokolle die TLS/SSL nutzen: https, ftps, imaps, pop3s, smtps, telnets, sips, ...
- Die eigentlichen Protokolldaten werden anstatt im eigenen Anwendungsprotokoll über das **Application Data- / Record Layer-Protokoll** von TLS/SSL verschlüsselt und integritätsgesichert übertragen (z.B. HTTP <-> HTTPS).

TLS/SSL-Technologie
→ Zusammenfassung (2/3)

- Anhand der **Portnummer** lässt sich erkennen, um welches ursprüngliche Protokoll es sich bei den übertragenen Daten handelt.
- TLS/SSL besteht aus zwei Schichten, nämlich dem Record Layer-Protokoll und den vier TLS/SSL-Teilprotokollen:
  - ChangeCipherSpec
  - Alert
  - Handshake
  - Application Data
- Der Protokollablauf bei TLS/SSL erfolgt in zwei Schritten:
  1. Schritt: Verbindungsaufbau, unterteilt in 4 Phasen:
  - 1. Phase: Aushandlung der Sicherheitsparameter.
  - 2. Phase: Serverauthentisierung (Optional) und Schlüsselaustausch
  - 3. Phase: Clientauthentisierung (Optional) und Schlüsselaustausch
  - 4. Phase: Beendigung des Handshakes

  2. Schritt: Transfer-Mode
  Verschlüsselte und *integritätsgesicherte* Datenübertragung

TLS/SSL-Technologie
→ Zusammenfassung (3/3)

- TLS/SSL beinhaltet 2 wichtige Instanzenkonzepte:
  - TLS/SSL-Session
  - TLS/SSL-Connection
- **Cipher Suites** sind eine Kombination aus **Schlüsselaustauschverfahren**, **Verschlüsselungsverfahren** mit **Schlüssellänge** und ein **Verfahren zum Integritätscheck**.
- Aufgabe eines Domänen-Zertifikats:
  Eindeutige Zuordnung eines Public-Key's zu einer Organisation.
- TLS/SSL bietet 3 Verbindungsarten:
  1. Server und Client ohne Authentifizierung
  2. Server authentifiziert, Client anonym
  3. Server und Client authentifiziert

Westfälische Hochschule
Gelsenkirchen Bocholt Recklinghausen
University of Applied Sciences

Transport Layer Security (TLS)
Secure Socket Layer (SSL)

Anwendungsunabhängiges Cyber-Sicherheitsprotokoll auf der Transport-Schicht

Fragen ?

Prof. Dr. (TU NN)
Norbert Pohlmann

Institut für Internet-Sicherheit – if(is)
Westfälische Hochschule, Gelsenkirchen
http://www.internet-sicherheit.de

if(is) internet-sicherheit.

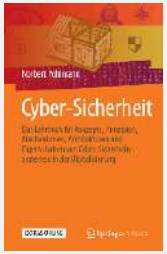
Anhang / Credits

Wir empfehlen

- **Kostenlose App securityNews**

securityNews

- **7. Sinn im Internet (Cyberschutzraum)**
  https://www.youtube.com/cyberschutzraum
- **Master Internet-Sicherheit**
  https://it-sicherheit.de/master-studieren/
- **Cyber-Sicherheit**
  Das **Lehrbuch** für Konzepte, Mechanismen, Architekturen und Eigenschaften von Cyber-Sicherheitssystemen in der Digitalisierung“, Springer Vieweg Verlag, Wiesbaden 2019
- https://norbert-pohlmann.com/cyber-sicherheit/



Besuchen und abonnieren Sie uns :-)

WWW
https://www.internet-sicherheit.de

Facebook
https://www.facebook.com/Internet.Sicherheit.ifis

Twitter
https://twitter.com/_ifis

YouTube
https://www.youtube.com/user/InternetSicherheitDE/

Prof. Norbert Pohlmann
https://norbert-pohlmann.com/

Quellen Bildmaterial

Eingebettete Piktogramme:
- Institut für Internet-Sicherheit – if(is)

Der Marktplatz IT-Sicherheit

(IT-Sicherheits-) Anbieter, Lösungen, Jobs, Veranstaltungen und Hilfestellungen (Ratgeber, IT-Sicherheitstipps, Glossar, u.v.m.) leicht & einfach finden.
https://www.it-sicherheit.de/