9.3 Merksätze und Prüfungswissen zu OSI-Schicht 6

Diese Seite fasst die wichtigsten Inhalte zur OSI-Schicht 6 zusammen.

OSI-Schicht 6 heißt:

```
Darstellungsschicht
```

Die Hauptaufgabe von Schicht 6 ist:

```
Daten so darstellen, codieren, komprimieren oder verschlüsseln,
dass sie vom Empfänger korrekt verstanden werden können.
```

Dabei sind besonders wichtig:

```
- Datenformate
- Zeichencodierung
- Kompression
- Verschlüsselung
- Entschlüsselung
- Zertifikate
- TLS
- Datenumwandlung
- Serialisierung
- MIME-Typen
```

Merksatz:

```
Schicht 6 = Darstellung, Codierung, Kompression und Verschlüsselung.
```

Grundidee von Schicht 6

Schicht 6 beschreibt, in welcher Form Daten vorliegen.

Beispiele:

Text als UTF-8
Bild als JPEG
Daten als JSON
Webseite über TLS verschlüsselt
Datei als ZIP komprimiert

Damit Kommunikation funktioniert, müssen Sender und Empfänger die Daten gleich interpretieren.

Merksatz:

Schicht 6 sorgt dafür,
dass Daten richtig verstanden werden.

Schicht 5, 6 und 7 unterscheiden

Schicht	Name	Grundidee
5	Sitzungsschicht	Sitzungen verwalten
6	Darstellungsschicht	Daten darstellen, codieren, schützen
7	Anwendungsschicht	Netzwerkdienste für Anwendungen

Beispiel Webanwendung:

Schicht 5:
Benutzer bleibt angemeldet.

Schicht 6:
Daten sind UTF-8-codiert und per TLS geschützt.

Schicht 7:
HTTP liefert Webseite oder API.

Merksatz:

Schicht 5 = Sitzung.
Schicht 6 = Darstellung.
Schicht 7 = Anwendung.

Darstellung bedeutet Datenrepräsentation

Darstellungsschicht bedeutet nicht nur:

> Anzeige auf dem Bildschirm

Sondern fachlich:

> Wie sind Daten codiert?
> In welchem Format liegen sie vor?
> Sind sie komprimiert?
> Sind sie verschlüsselt?
> Wie werden sie interpretiert?

Merksatz:

> Darstellungsschicht meint Datenrepräsentation,
> nicht nur optische Anzeige.

Datenformat

Ein Datenformat legt fest, wie Daten aufgebaut sind.

Beispiele:

Datenart	Beispiele
Text / Struktur	JSON, XML, HTML
Bild	JPEG, PNG, GIF, SVG
Audio	WAV, MP3, AAC, FLAC
Video	MP4, H.264, H.265
Dokument	PDF, DOCX
Archiv	ZIP, 7z

Wenn der Empfänger das Format nicht versteht, kann er die Daten nicht korrekt verarbeiten.

Merksatz:

> Datenformat = Aufbau und Struktur von Daten.

JSON

JSON steht für:

```
JavaScript Object Notation
```

JSON ist ein häufiges Format für strukturierte Daten.

Typisch bei:

```
- APIs
- Webanwendungen
- Konfigurationsdateien
- Datenaustausch zwischen Systemen
```

Beispiel:

```
{
  "name": "Felix",
  "rolle": "FISI",
  "aktiv": true
}
```

Merksatz:

```
JSON ist ein kompaktes Format für strukturierte Daten.
```

XML

XML steht für:

```
Extensible Markup Language
```

XML ist ein strukturiertes Textformat mit Tags.

Typisch bei:

```
- älteren Schnittstellen
- Konfigurationsdateien
- Dokumentenformaten
- Webservices
- Datenaustausch
```

Beispiel:

```
<benutzer>
  <name>Felix</name>
  <rolle>FISI</rolle>
</benutzer>
```

Merksatz:

```
XML nutzt Tags zur Strukturierung von Daten.
```

JSON und XML vergleichen

Merkmal	JSON	XML
Schreibweise	kompakter	ausführlicher
Struktur	Objekte und Arrays	Tags und Elemente
häufig bei	moderne Web-APIs	ältere Schnittstellen, Dokumente
Lesbarkeit	meist kürzer	oft länger
Datenaustausch	sehr verbreitet	weiterhin wichtig

Merksatz:

```
JSON und XML sind Formate für strukturierten Datenaustausch.
```

Zeichencodierung

Zeichencodierung legt fest, wie Zeichen als digitale Werte gespeichert werden.

Beispiele für Zeichen:

```
A
ä
ß
€
□
□□
```

Ohne passende Codierung können Zeichen falsch dargestellt werden.

Merksatz:

```
Zeichencodierung übersetzt Zeichen in digitale Werte.
```

ASCII

ASCII ist eine ältere Zeichencodierung.

ASCII enthält vor allem:

```
- englische Buchstaben
- Ziffern
- Satzzeichen
- Steuerzeichen
```

Klassisches ASCII enthält keine deutschen Umlaute wie:

```
ä
ö
ü
ß
```

Merksatz:

```
ASCII ist einfach,
aber für viele Sprachen zu begrenzt.
```

Unicode

Unicode ist ein Standard für sehr viele Zeichen.

Dazu gehören:

```
- lateinische Zeichen
- deutsche Umlaute
- griechische Zeichen
- kyrillische Zeichen
- asiatische Schriftzeichen
- Symbole
- Emojis
```

Unicode beschreibt, welche Zeichen es gibt.

Merksatz:

```
Unicode ist ein großer Zeichensatzstandard.
```

UTF-8

UTF-8 ist eine sehr verbreitete Codierung für Unicode.

UTF-8 wird häufig verwendet bei:

```
- Webseiten
- Linux-Systemen
- APIs
- Datenbanken
- Konfigurationsdateien
- E-Mails
```

Vorteil:

```
UTF-8 kann sehr viele Zeichen darstellen
und ist mit einfachem ASCII gut kompatibel.
```

Merksatz:

```
UTF-8 ist die wichtigste moderne Zeichencodierung.
```

Typischer Codierungsfehler

Ein typischer Fehler ist:

```
Müller
```

wird angezeigt als:

```
MÃ¼ller
```

Das deutet häufig darauf hin, dass UTF-8-Daten mit falscher Codierung interpretiert wurden.

Mögliche Ursachen:

```
- falsche Dateicodierung
- falsche Datenbankcodierung
- falscher HTTP-Header
- falscher Import
- falsche Anwendungseinstellung
```

Merksatz:

```
Kaputte Umlaute sind oft Codierungsprobleme.
```

Kompression

Kompression bedeutet:

```
Daten werden verkleinert.
```

Ziele:

```
- weniger Speicherplatz
- weniger Datenverkehr
- schnellere Übertragung
- geringere Bandbreitennutzung
```

Beispiele:

```
ZIP
gzip
Brotli
JPEG
MP3
H.264
```

Merksatz:

```
Kompression reduziert die Datenmenge.
```

Verlustfreie Kompression

Verlustfreie Kompression bedeutet:

```
Die Originaldaten können vollständig wiederhergestellt werden.
```

Beispiele:

```
- ZIP
- gzip
- PNG
- FLAC
```

Typisch für:

```
- Text
- Dokumente
- Programme
- Datenbanken
- Quellcode
```

Merksatz:

```
Verlustfrei = exakt wiederherstellbar.
```

Verlustbehaftete Kompression

Verlustbehaftete Kompression bedeutet:

```
Daten werden verkleinert,
aber ein Teil der ursprünglichen Information geht verloren.
```

Beispiele:

```
- JPEG
- MP3
- AAC
- H.264
- H.265
```

Typisch für:

```
- Fotos
- Musik
- Videos
```

Merksatz:

```
Verlustbehaftet = kleiner,
aber nicht exakt original.
```

Codierung, Kompression und Verschlüsselung unterscheiden

Begriff	Ziel	Sicherheitsfunktion
Codierung	Daten in anderes Format bringen	nein
Kompression	Datenmenge reduzieren	nein
Verschlüsselung	Daten vor unbefugtem Lesen schützen	ja

Wichtig:

```
Base64 ist Codierung,
aber keine Verschlüsselung.

ZIP ist Kompression,
aber ohne Passwort keine Verschlüsselung.
```

Merksatz:

```
Codierung ist nicht automatisch Sicherheit.
```

Verschlüsselung

Verschlüsselung bedeutet:

```
Daten werden so umgewandelt,
dass sie ohne passenden Schlüssel nicht lesbar sind.
```

Ziel:

```
Schutz der Vertraulichkeit
```

Beispiele:

```
- HTTPS
- TLS
- VPN
- E-Mail-Verschlüsselung
- Festplattenverschlüsselung
```

Merksatz:

```
Verschlüsselung schützt vor unbefugtem Lesen.
```

Entschlüsselung

Entschlüsselung bedeutet:

```
Verschlüsselte Daten werden mit dem passenden Schlüssel wieder lesbar gemacht.
```

Ohne passenden Schlüssel bleiben die Daten unverständlich.

Merksatz:

```
Entschlüsselung macht verschlüsselte Daten wieder lesbar.
```

TLS

TLS steht für:

```
Transport Layer Security
```

TLS schützt Daten bei der Übertragung.

Wichtige Schutzziele:

```
- Vertraulichkeit
- Integrität
- Authentizität
```

Das bekannteste Beispiel ist:

```
HTTPS
```

Merksatz:

```
TLS schützt Daten zwischen Client und Server.
```

HTTPS

HTTPS bedeutet:

```
HTTP über TLS
```

Vergleich:

Protokoll	Verschlüsselung	typischer Port
HTTP	nein	TCP 80
HTTPS	ja, TLS	TCP 443

Merksatz:

```
HTTPS = HTTP mit TLS-Schutz.
```

TLS im OSI-Modell

TLS wird häufig der Darstellungsschicht zugeordnet, weil es Daten verschlüsselt, entschlüsselt und schützt.

In der Praxis liegt TLS zwischen Anwendung und Transport.

Vereinfacht:

```
HTTP
TLS
TCP
IP
Ethernet
```

Merksatz:

```
TLS wird oft Schicht 6 zugeordnet,
hat aber praktischen Bezug zu Schicht 4 und 7.
```

TLS ist nicht TCP

TCP ist ein Transportprotokoll.

TLS ist eine Schutzschicht für die übertragenen Daten.

Unterschied:

Thema	Einordnung
TCP	Transport auf Schicht 4
TLS	Verschlüsselung und Schutz

Thema	Einordnung
HTTP	Anwendung auf Schicht 7
HTTPS	HTTP über TLS

Merksatz:

```
TCP transportiert.
TLS schützt.
HTTP verwendet.
```

Zertifikat

Ein Zertifikat ist ein digitaler Vertrauensnachweis.

Es hilft dem Client zu prüfen:

```
Spreche ich mit dem richtigen Server?
```

Ein Zertifikat enthält unter anderem:

```
- Domainname
- öffentlicher Schlüssel
- Aussteller
- Gültigkeitszeitraum
- Signatur
- alternative Namen
```

Merksatz:

```
Zertifikat = digitaler Vertrauensnachweis.
```

Zertifizierungsstelle

Eine Zertifizierungsstelle wird genannt:

```
Certificate Authority
```

Kurz:

```
CA
```

Die CA stellt Zertifikate aus und signiert sie.

Der Client vertraut einem Zertifikat, wenn er der ausstellenden CA oder der Zertifikatskette vertraut.

Merksatz:

```
CA = vertrauenswürdige Stelle für Zertifikate.
```

Zertifikatskette

Eine Zertifikatskette verbindet das Serverzertifikat mit einer vertrauenswürdigen Root-CA.

Typische Bestandteile:

```
- Serverzertifikat
- Zwischenzertifikat
- Root-Zertifikat
```

Wenn ein Zwischenzertifikat fehlt, kann es zu Zertifikatswarnungen kommen.

Merksatz:

```
Zertifikatskette muss vollständig und vertrauenswürdig sein.
```

Domainname im Zertifikat

Das Zertifikat muss zum aufgerufenen Namen passen.

Beispiel:

```
Aufgerufen:
wiki.firma.de
```

Das Zertifikat muss für diesen Namen gültig sein.

Wenn das Zertifikat für einen anderen Namen gilt, zeigt der Browser eine Warnung.

Merksatz:

```
Aufgerufener Name muss im Zertifikat enthalten sein.
```

Gültigkeitszeitraum

Zertifikate sind nur für einen bestimmten Zeitraum gültig.

Wenn ein Zertifikat abgelaufen ist, zeigen Clients Warnungen oder blockieren die Verbindung.

Auch falsche Systemzeit kann Zertifikatsfehler erzeugen.

Merksatz:

```
Zertifikat und Systemzeit müssen gültig sein.
```

Selbstsigniertes Zertifikat

Ein selbstsigniertes Zertifikat wurde nicht von einer allgemein vertrauenswürdigen CA signiert.

Es kann technisch verschlüsseln, ist aber für Clients nicht automatisch vertrauenswürdig.

Typische Nutzung:

```
- Testumgebung
- Labor
- Homelab
- interne Systeme
```

Merksatz:

```
Selbstsigniert kann verschlüsseln,
ist aber nicht automatisch vertrauenswürdig.
```

Öffentlicher und privater Schlüssel

Bei asymmetrischer Kryptografie gibt es:

```
öffentlichen Schlüssel
privaten Schlüssel
```

Der öffentliche Schlüssel darf bekannt sein.

Der private Schlüssel muss geheim bleiben.

Bei TLS steckt der öffentliche Schlüssel im Zertifikat.

Merksatz:

```
Öffentlicher Schlüssel darf öffentlich sein.
Privater Schlüssel muss geheim bleiben.
```

TLS-Handshake

Beim TLS-Handshake wird die sichere Verbindung vorbereitet.

Vereinfacht:

```
1. Client verbindet sich mit Server.
2. Server sendet Zertifikat.
3. Client prüft Zertifikat.
4. Verfahren werden ausgehandelt.
5. Sitzungsschlüssel werden gebildet.
6. Nutzdaten werden verschlüsselt übertragen.
```

Merksatz:

```
TLS-Handshake bereitet verschlüsselte Kommunikation vor.
```

TCP-Handshake und TLS-Handshake unterscheiden

Bei HTTPS passiert vereinfacht:

```
1. TCP-Handshake
2. TLS-Handshake
3. HTTP-Kommunikation
```

Unterschied:

Vorgang	Zweck
TCP-Handshake	Transportverbindung aufbauen
TLS-Handshake	sichere Verbindung aushandeln
HTTP-Kommunikation	Anwendungskommunikation

Merksatz:

```
Erst TCP,
dann TLS,
dann HTTP.
```

Schutzziele von TLS

Schutzziel	Bedeutung
Vertraulichkeit	Schutz vor Mitlesen
Integrität	Schutz vor unbemerkter Veränderung
Authentizität	Prüfung der Gegenstelle

Merksatz:

```
TLS schützt Vertraulichkeit,
Integrität
und Authentizität.
```

Base64

Base64 ist eine Codierung.

Sie wandelt Binärdaten in Textzeichen um.

Typische Nutzung:

```
- E-Mail-Anhänge
- Tokens
- Zertifikatsdaten
- API-Daten
```

Wichtig:

```
Base64 ist leicht rückgängig zu machen.
Base64 ist keine Verschlüsselung.
```

Merksatz:

```
Base64 = Codierung,
nicht Verschlüsselung.
```

Hashing

Hashing ist keine Verschlüsselung.

Ein Hash ist ein Prüfwert.

Beispiele:

```
- SHA-256
- SHA-512
```

Eigenschaften:

```
- gleiche Eingabe ergibt gleichen Hash
- kleine Änderung ergibt anderen Hash
- Einwegfunktion
- Original soll nicht aus Hash zurückgerechnet werden
```

Merksatz:

```
Hashing prüft Integrität,
verschlüsselt aber nicht.
```

Passwort-Hashing

Passwörter sollten nicht im Klartext gespeichert werden.

Auch normale schnelle Hashfunktionen sind für Passwörter oft nicht ideal.

Für Passwörter nutzt man spezielle Verfahren mit Salt und Arbeitsfaktor.

Beispiele:

```
- bcrypt
- Argon2
- PBKDF2
```

Merksatz:

```
Passwörter sicher hashen,
nicht im Klartext speichern.
```

Serialisierung

Serialisierung bedeutet:

```
Datenstrukturen werden in ein übertragbares oder speicherbares Format umgewandelt.
```

Beispiel:

```
Objekt in Anwendung
→ JSON
→ Übertragung
→ JSON wird beim Empfänger wieder verarbeitet
```

Merksatz:

```
Serialisierung macht Daten übertragbar.
```

Deserialisierung

Deserialisierung ist die Rückumwandlung aus einem übertragenen Format.

Beispiel:

```
JSON
→ Objekt in Anwendung
```

Wichtig:

```
Unsichere Deserialisierung kann gefährlich sein,
wenn fremde Daten ungeprüft verarbeitet werden.
```

Merksatz:

```
Deserialisierung macht Daten wieder nutzbar.
```

MIME-Type

Ein MIME-Type beschreibt, welche Art von Daten übertragen wird.

Beispiele:

MIME-Type	Bedeutung
text/html	HTML-Dokument
application/json	JSON-Daten
image/png	PNG-Bild
image/jpeg	JPEG-Bild
text/css	CSS-Datei
application/pdf	PDF-Dokument

Merksatz:

```
MIME-Type sagt dem Client,
wie Daten zu interpretieren sind.
```

Typische Schicht-6-Fehler

Typische Fehler auf Schicht 6 sind:

```
- falsche Zeichencodierung
- falscher MIME-Type
- ungültiges JSON
- ungültiges XML
- falsches Dateiformat
- defekte Kompression
- nicht unterstütztes Kompressionsverfahren
- Zertifikat abgelaufen
- Zertifikat passt nicht zum Namen
- Zertifikatskette unvollständig
- TLS-Version zu alt
- Cipher Suite nicht kompatibel
- Base64 mit Verschlüsselung verwechselt
```

Merksatz:

```
Schicht-6-Fehler betreffen Darstellung,
Format,
Codierung
oder Schutz der Daten.
```

Fehlerbild: Komische Zeichen

Beispiel:

```
Müller
```

wird angezeigt als:

```
MÃ¼ller
```

Mögliche Ursachen:

```
- falsche Zeichencodierung
- falsche Datenbankcodierung
- falscher HTTP-Header
- falscher Import
```

- Datei falsch gespeichert

Merksatz:

Komische Zeichen = oft Codierungsproblem.

Fehlerbild: HTTPS-Warnung

Mögliche Ursachen:

- Zertifikat abgelaufen
- Domainname passt nicht
- CA nicht vertrauenswürdig
- Zwischenzertifikat fehlt
- selbstsigniertes Zertifikat
- Systemzeit falsch
- Zertifikat wurde widerrufen

Merksatz:

HTTPS-Warnung = Zertifikat und TLS prüfen.

Fehlerbild: API versteht Daten nicht

Mögliche Ursachen:

- ungültiges JSON
- falscher Content-Type
- falsche Zeichencodierung
- falsche Datentypen
- fehlende Pflichtfelder
- falsches Datumsformat
- ungültige Serialisierung

Merksatz:

API-Fehler können Format- oder Darstellungsprobleme sein.

Fehlerbild: Datei lässt sich nicht öffnen

Mögliche Ursachen:

- falsches Dateiformat
- falsche Dateiendung
- beschädigte Datei
- falscher MIME-Type
- fehlender Codec
- nicht unterstützte Kompression
- Datei unvollständig übertragen

Merksatz:

Datei-Probleme können Format- oder Codierungsprobleme sein.

Schicht 6 und Sicherheit

Schicht 6 hat starken Sicherheitsbezug.

Wichtige Themen:

- TLS
- Zertifikate
- Verschlüsselung
- Integrität
- sichere Codierung
- sichere Deserialisierung
- Hashing
- Schutz vor unsicherer Datenverarbeitung

Merksatz:

Falsche Datenverarbeitung kann Sicherheitslücken erzeugen.

Was Schicht 6 nicht macht

Schicht 6 macht nicht:

- Bits über Kabel übertragen
- MAC-Adressen verwenden
- IP-Pakete routen
- TCP-Verbindungen aufbauen
- Ports öffnen
- Sitzungen verwalten
- DNS-Namen auflösen
- konkrete Anwendung bereitstellen

Diese Aufgaben liegen auf anderen Schichten.

Merksatz:

Schicht 6 stellt Daten dar,
transportiert sie aber nicht selbst.

Prüfungswissen: wichtigste Zuordnungen

Begriff	richtige Einordnung
UTF-8	Schicht 6
ASCII	Schicht 6
Unicode	Schicht 6
JSON	Schicht 6 / 7-Bezug
XML	Schicht 6 / 7-Bezug
Kompression	Schicht 6
Verschlüsselung	Schicht 6
TLS	häufig Schicht 6, praktisch zwischen 4 und 7
Zertifikat	Schicht 6 / 7-Bezug
MIME-Type	Schicht 6 / 7-Bezug
Base64	Codierung, Schicht 6
Hashing	Sicherheits-/Integritätsbezug
TCP	Schicht 4
HTTP	Schicht 7
DNS	Schicht 7

Merksatz:

Codierung,
Kompression
und Verschlüsselung
sind typische Schicht-6-Themen.

Typische IHK-Fragen

In AP1 und AP2 kann zum Beispiel gefragt werden:

- Welche Aufgabe hat die Darstellungsschicht?
- Was bedeutet Datenformat?
- Was ist Zeichencodierung?
- Was ist UTF-8?
- Warum werden Umlaute manchmal falsch angezeigt?
- Was ist Kompression?
- Was ist der Unterschied zwischen verlustfrei und verlustbehaftet?
- Was ist Verschlüsselung?
- Was ist TLS?
- Was ist HTTPS?
- Warum ist Base64 keine Verschlüsselung?
- Was ist ein Zertifikat?
- Warum muss der Name zum Zertifikat passen?
- Was ist eine Zertifizierungsstelle?
- Was ist eine Zertifikatskette?
- Warum kann eine HTTPS-Warnung erscheinen?
- Was ist ein MIME-Type?
- Was ist Serialisierung?

Typische Prüfungsfallen

Schicht 6 heißt Darstellungsschicht.

Darstellung bedeutet Datenrepräsentation.

Schicht 6 ist nicht nur Bildschirmdarstellung.

UTF-8 ist eine Zeichencodierung.

ASCII ist begrenzt.

Unicode ist der Zeichensatzstandard.

JSON und XML sind Datenformate.

Kompression ist keine Verschlüsselung.

Codierung ist keine Verschlüsselung.

Base64 ist keine Verschlüsselung.

Hashing ist keine Verschlüsselung.

TLS schützt Daten bei der Übertragung.

HTTPS nutzt TLS.

TLS ist nicht TCP.

TLS ist nicht HTTPS.

Zertifikat muss zum Domainnamen passen.

Zertifikat muss gültig sein.

Client muss der CA vertrauen.

Fehlende Zwischenzertifikate können Warnungen verursachen.

Falsche Systemzeit kann Zertifikatsfehler verursachen.

MIME-Type hilft bei der Interpretation von Daten.

Schicht 6 ist in der Praxis oft eng mit Schicht 7 verbunden.

Wichtige Begriffe kurz erklärt

Begriff	Kurze Erklärung
Darstellungsschicht	OSI-Schicht 6
Datenformat	Aufbau und Struktur von Daten
Zeichencodierung	Zuordnung von Zeichen zu digitalen Werten
ASCII	ältere einfache Zeichencodierung
Unicode	Standard für viele Zeichen weltweit
UTF-8	verbreitete Unicode-Codierung
Kompression	Verkleinerung von Daten
verlustfrei	Original exakt wiederherstellbar
verlustbehaftet	Datenverlust zugunsten kleinerer Größe
Verschlüsselung	Daten unlesbar für Unbefugte machen
Entschlüsselung	verschlüsselte Daten wieder lesbar machen
TLS	Schutzschicht für Übertragung
HTTPS	HTTP über TLS
Zertifikat	digitaler Vertrauensnachweis
CA	Zertifizierungsstelle
Zertifikatskette	Vertrauenskette von Serverzertifikat zu Root-CA
MIME-Type	Angabe des Medientyps
Base64	Codierung von Binärdaten als Text
Hashing	Bildung eines Einweg-Prüfwerts
Serialisierung	Umwandlung in übertragbares Format
Deserialisierung	Rückumwandlung aus übertragbarem Format

IHK-sichere Gesamtformulierung

Die Darstellungsschicht ist Schicht 6 des OSI-Modells. Sie sorgt dafür, dass Daten in einer geeigneten Form dargestellt, codiert, komprimiert, verschlüsselt oder umgewandelt werden, damit der Empfänger sie korrekt interpretieren kann. Typische Themen sind Zeichencodierung wie UTF-8, Datenformate wie JSON oder XML, Kompression wie gzip oder ZIP, Verschlüsselung mit TLS sowie Zertifikate bei HTTPS. TLS wird häufig der Darstellungsschicht zugeordnet, liegt in der Praxis aber zwischen Anwendung und Transport. Codierung, Kompression und Verschlüsselung sind voneinander zu unterscheiden.

Wichtigste Merksätze

Schicht 6 = Darstellungsschicht.

Schicht 6 = Daten darstellen,
codieren,
komprimieren
und verschlüsseln.

Darstellung bedeutet Datenrepräsentation.

Datenformat legt den Aufbau von Daten fest.

JSON und XML sind Datenformate.

Zeichencodierung übersetzt Zeichen in digitale Werte.

ASCII ist begrenzt.

Unicode umfasst viele Zeichen.

UTF-8 ist sehr verbreitet.

Falsche Codierung erzeugt Zeichenfehler.

Kompression reduziert Datenmenge.

Verlustfrei = exakt wiederherstellbar.

Verlustbehaftet = kleiner,
aber nicht exakt original.

Verschlüsselung schützt Vertraulichkeit.

Entschlüsselung macht Daten wieder lesbar.

TLS schützt Daten bei der Übertragung.

HTTPS = HTTP über TLS.

TLS ist nicht TCP.

TLS ist nicht HTTPS.

TCP transportiert.

TLS schützt.

HTTP verwendet.

Zertifikat = Vertrauensnachweis.

CA = Zertifizierungsstelle.

Zertifikatskette muss vollständig sein.

Domainname muss zum Zertifikat passen.

Zertifikat muss gültig sein.

Base64 ist Codierung,
keine Verschlüsselung.

Hashing ist keine Verschlüsselung.

MIME-Type beschreibt den Datentyp.

Serialisierung macht Daten übertragbar.

Schicht 6 ist oft eng mit Schicht 7 verbunden.