# IT Sicherheit:
# Netzwerksicherheit: SSL/TLS

Dr. Christian Rathgeb

Hochschule Darmstadt, CRISP, da/sec Security Group

15.05.2018

# Netzwerke: Einführung I

- *Computernetzwerk:* Zwei oder mehrere Computer, die durch ein Übertragungsmedium miteinander verbunden (vernetzt) sind, bilden ein Computernetzwerk
- Ein Computernetz besteht daher mindestens aus zwei Knoten (Rechner) und einer (physikalischen + logischen) Verbindung
- Beispiele: Internet, Computernetzwerke, Mobilfunknetzwerke, Optische Netzwerke
- Fokus der IT-Sicherheit: Datennetzwerke (Internet-Protokoll (IP) Netzwerke)

# Netzwerke: Einführung II

- Arten der Vernetzung:
  1. Wireless-LAN
  2. Ethernetkabel
  3. Bluetooth
  4. Near Field Communication (NFC)
  5. Mobilfunk (UMTS, LTE)
  6. Optische-Links (Glasfaser)
  7. ...

# Netzwerke: Einführung III

Sinn und Zweck von Computernetzwerken:

- Zugriffe auf Daten, Programme, Ressourcen anderer Rechner ist möglich, z.B.
  - Zugriff auf Hochleistungsrechner
  - Abruf von Filmen aus einem Archiv
- Verteilung von Rechenleistung und Datenhaltung auf unterschiedliche Rechner
  - Erhöhte Flexibilität und Ausfallsicherheit
- Rechnergestützte Aufgaben können arbeitsteilig ausgeführt werden
  - Verteilung von Rechenaufgaben auf unterschiedliche Computer

# Netzwerke: Einführung IV

Bausteine eines Computernetzwerkes:

- *Endgeräte/„Computer“:* Laptop, PC, Smartphone, Webserver
- *Hardware für die physikalische Übertragung:* Verkabelung der Netzwerkgeräte (Router, Switches, ...)
- *Netzwerk-Software:* Implementierung von Protokollen
- *Netzwerk-Applikationen:* Web, Email, etc.

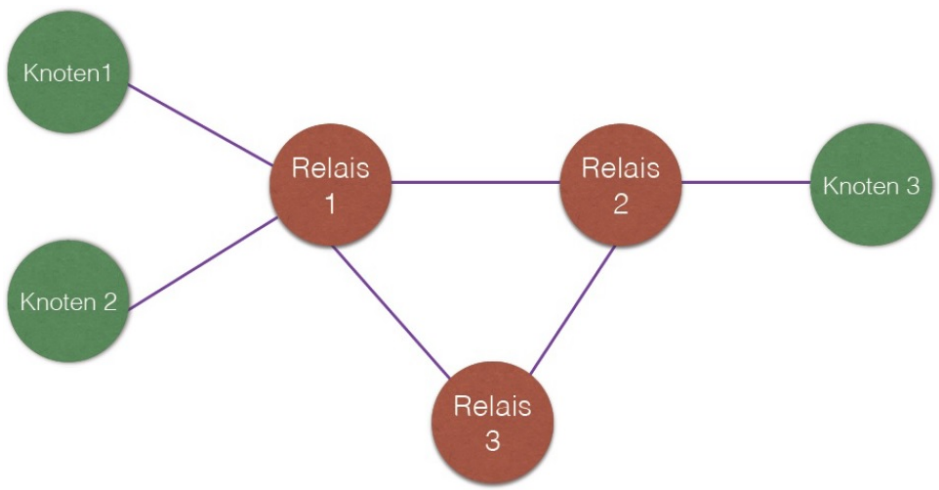
# Netzwerke: Einführung V

Eine direkte physikalische Verbindung zwischen allen Knoten eines Netztwerkes ist in der Realität nicht umsetzbar!

- zu viele Verbindungen von jedem Knoten aus; Entfernungen zwischen Knoten

Computernetztwerke haben daher **Relais-Knoten**

- Knoten sind nicht alle direkt miteinander verbunden, Relais-Knoten vermitteln zwischen Knotengruppen



# Netzwerke: Einführung VI

Netzwerkmethoden:

1. *Naming:* Ein Name für einen Knoten oder einen Dienst (Knoten $1$, Knoten $2$, ..., Knoten $n$)
2. *Addressing:* Eine Adresse für einen Knoten oder Dienst (Wie ist die „Anschrift“ eines Knotens?)

- Kann man mit der Adresse alleine Daten zwischen den Knoten austauschen?

# Netzwerke: Einführung VI

Netzwerkmethoden:

1. *Naming:* Ein Name für einen Knoten oder einen Dienst (Knoten $1$, Knoten $2$, ..., Knoten $n$)
2. *Addressing:* Eine Adresse für einen Knoten oder Dienst (Wie ist die „Anschrift“ eines Knotens?)

- Kann man mit der Adresse alleine Daten zwischen den Knoten austauschen?
- Nein, denn es fehlt die Information welchen Pfad die Daten durch das Netzwerk nehmen müssen

# Netzwerke: Einführung VII

Netzwerkmethoden:

3. *Routing:* Bestimmung des Pfades den die Daten durch das Netzwerk nehmen

- Datenaustausch zwischen beliebigen Konten erfordert die Wahl eines Pfades durch das Netzwerk
- Relais müssen Routen zu den anderen Relais und Knoten kennen $\rightarrow$ Routingtabellen beinhalten diese Information

4. *Forwarding:* Weiterleiten der Daten von einem Netzwerksegment in das nächste

- Versand/Weiterleitung (engl. Forwarding) ist der Prozess des Datentransfer mit den Information aus dem Routing

# Das Internet

- Verbindung von mehreren unterschiedlichen Netzwerktechnologien zu einem grossen Netzwerk (INTER Networking) $\rightarrow$ ein Netz aus Netzen!
- Millionen vernetzter Computer: Hosts = Endsysteme auf denen Netzwerk-Applikationen laufen
- Verbunden durch Leitungen oder Funkstrecken: Glasfaser, Kupfer, Funk
- Vermittlungsstellen = Router (leiten Datenpakete weiter)

# (Kommunikations-)Protokolle

- Ein Protokoll ist eine eindeutig definierte Abfolge von Handlungen zwischen Kommunikationspartnern
- Protokoll setzt sich aus mehreren Schritten zusammen
- Voraussetzungen:
  1. Jede/r muss alle Schritte kennen
  2. Jede/r muss zustimmen den Schritten zu folgen
  3. Protokoll ist eindeutig
  4. Für jede Situation gibt es einen definierten Schritt
- Weiters legt ein Kommunikations-Protokoll das Format der Nachrichten (Syntax) fest!

# Das menschliche Telefonprotokoll

Telefonanruf folgt einigen Regeln (wenn man höflich ist):

1. Anrufer wählt die Telefonnummer und wartet auf das Wählzeichen
2. Anrufer wartet auf das Abnehmen des Angerufenen
3. Angerufenen hebt ab und nennt seinen Namen
4. Anrufer wartet ab bis der Angerufene seinen Satz zu Ende gesagt hat und meldet sich ebenfalls mit seinem Namen.
5. Danach beginnt die eigentliche Unterhaltung
6. ...

# Schichtenmodelle

Netzwerke sind komplexe Gebilde!

- Unterschiedliche Dienste, Knotenarten, Protokolle, etc
- Komplexität verlangt die Aufteilung in Funktionsblöcke oder Schichten
- Schichten sind eine Sammlung gleicher Funktionalität
- Beherrschung der Komplexität in einer Schicht einfacher
- Entwicklung der Schichten mit Ihrer Funktionalität teilweise getrennt von anderen Schichten
- Grundprinzip: “Teile und Herrsche!”

# DoD-Schichtenmodell I

Auf dem DoD-Schichtenmodell (Abk. DoD: Department of Defense - Verteidigungsministerium) basiert das Internet. Es besteht aus insgesamt vier Schichten:

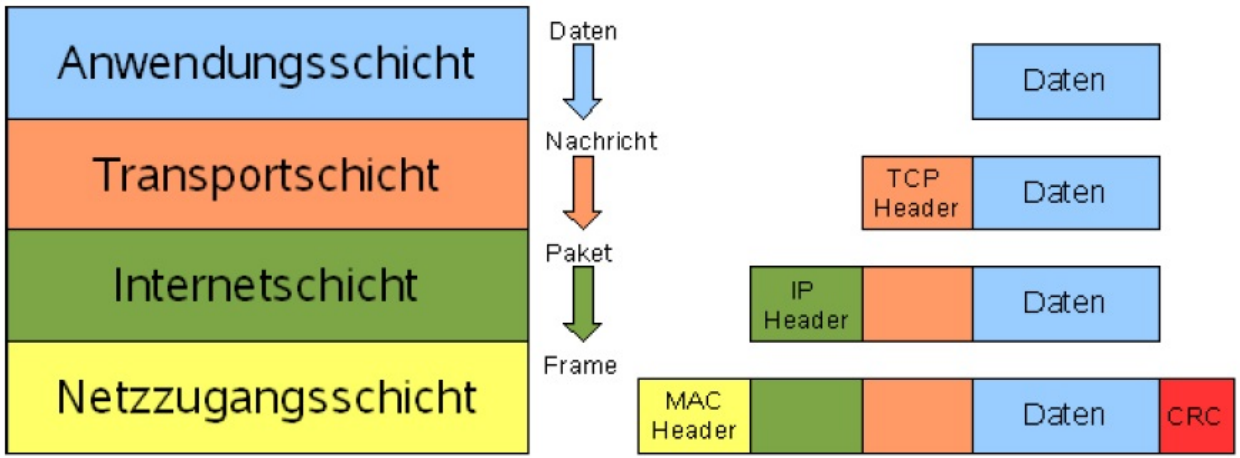
1. *Anwendungsschicht - Application Layer:* Applikationen, die über das Internet miteinander kommunizieren (zB: HTTP, FTP, SMTP)
2. *Transportschicht - Transport Layer:* Schicht zur Ermöglichung von gesichertem Datentransport mit Flußkontrolle; keine Überflutung des Empfängers, Wiederholung der Sendung bei Timeout, zuverlässiger Bytestrom, (zB: TCP (Transmission Control Protocol), UDP (User Datagram Protocol)).

# DoD-Schichtenmodell II

Auf dem DoD-Schichtenmodell (Abk. DoD: Department of Defense - Verteidigungsministerium) basiert das Internet. Es besteht aus insgesamt vier Schichten:

3. *Internetschicht - Internet Layer:* Schicht zur Erzeugung und Versendung der Datenpakete mit Hilfe der IP-Adresse des Absenders und Empfängers, Wegsteuerung eines Paketes (zB: Internet Protokoll IP)
4. *Netzzugangsschicht - Network Access Layer:* Schicht für die Datenübertragung von direkt miteinander verbundenen Rechnern unter Berücksichtigung der Auflösung einer logischen IP-Adresse in eine MAC-Adresse (MAC - Media Access Control - Kennung der Netzkarte) (zB Ethernet)

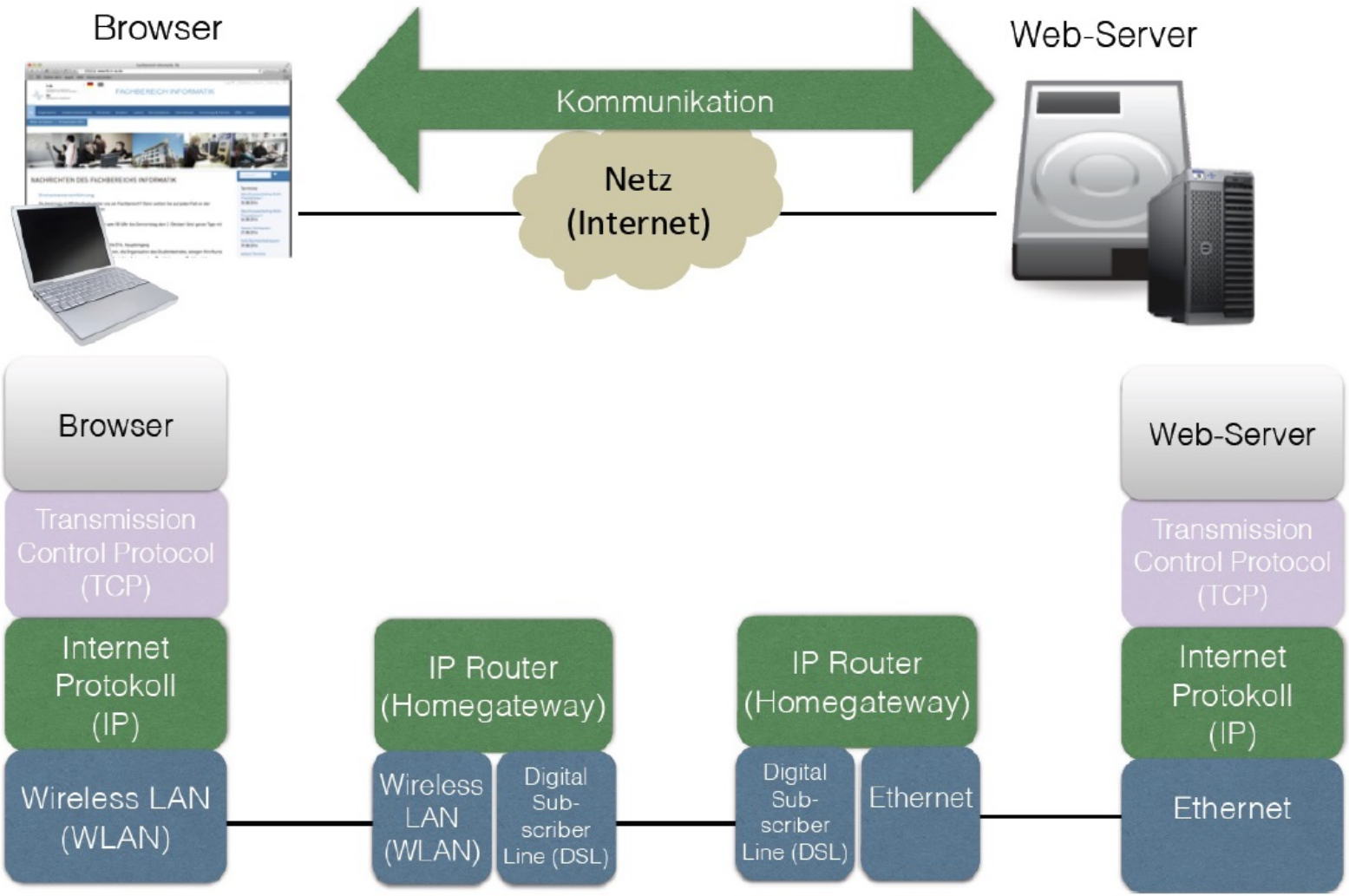
# DoD-Schichtenmodell III




Beispiel: Kommunikation zwischen Browser und Webserver

- nicht direkt zwischen Browser und Webserver!
- sondern über das Netzwerk!
- Netzwerk besteht aus unterschiedlichen Netzwerktechnologien: Wireless LAN (WLAN), Digital Subscriber Line (DSL), Kabelnetze, Mobilfunk (UMTS, LTE) etc.

# DoD-Schichtenmodell IV



# OSI-Schichtenmodell I

Das OSI-Schichtenmodell (Abk. OSI: open system interconnections) besteht in der standardisierten Fassung seit 1983 und dient mit sieben Schichten wesentlich genauer als das DoD-Schichtenmodell

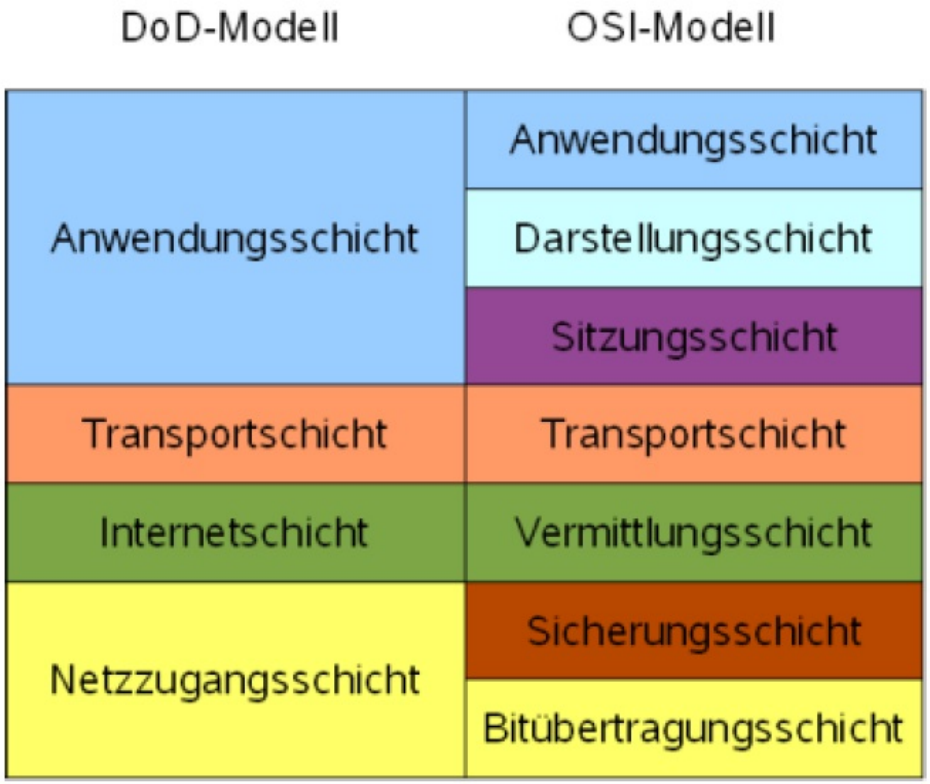
1. *Anwendungsschicht:* stellt den Anwendungen verschiedene Funktionalitäten zur Verfügung
2. *Darstellungsschicht:* wandelt systemabhängige Datendarstellung in eine unabhängige Form um, sorgt für Datenkompression und Verschlüsselung
3. *Sitzungsschicht:* sorgt für die Prozesskommunikation zwischen zwei Systemen, behandelt Sitzungsabbrüche

# OSI-Schichtenmodell II

4. *Transortschicht:* sorgt für die Zerlegung in Datenpaketen und die Stauvermeidung
5. *Vermittlungsschicht:* sorgt für die Weitervermittlung von Datenpaketen einschließlich der Wegsuche (Routing), Netzadressen
6. *Sicherungsschicht:* gewährleistet eine weitgehend fehlerfreie Übertragung, regelt den Zugriff auf das Übertragungsmedium
7. *Bitübertragungsschicht:* stellt mechanische, elektrische und weitere funktionale Hilfsmittel zur Verfügung, um physikalische Verbindungen zu aktivieren bzw. deaktivieren, sie aufrechtzuerhalten und Bits darüber zu übertragen

# OSI-Schichtenmodell III

Vergleich der beiden Schichtenmodelle:



# SSL/TLS: Einführung I

- Im Jahr 1994 veröffentlichte Netscape die erste Version des Protokolls Secure Sockets Layer (SSL) zum sicheren Aufbau von HTTP Verbindungen
- Im Verlauf der Zeit adaptierte die Internet Engineering Task Force (IETF) SSL als Internet Standard für beliebige Protokolle der Anwendungsschicht zwischen einem anfragenden Client und einem angefragten Server
- Dieser Standard wird mittlerweile von der IETF als Transport Layer Security (TLS) aktiv weiterentwickelt
- Die aktuelle Version ist TLS 1.2, das als Request for Comments 5246 standardisiert ist
- Kernkonzepte von TLS und SSL sind identisch!

# SSL/TLS: Einführung II

- Mit TLS 1.0 hat sich ein de facto Internet Standard für die Absicherung von Protokollen der Anwendungsschicht zur Absicherung einer Client-Server- Kommunikation etabliert
- Das gilt insbesondere für HTTP-Verbindungen, da TLS von gängigen Webbrowsern unterstützt wird
- Dabei fügt TLS eine weitere Schicht zwischen die OSI-Schichten Transport und Sitzung ein
- Sofern eine HTTP-Verbindung mit SSL bzw. TLS abgesichert ist, wird von „HTTP over TLS“ oder auch kurz HTTPS gesprochen

# SSL/TLS: Einführung III

- Zuerst findet zunächst eine geschützte Identifikation und Authentifizierung der Kommunikationspartner statt.
- Anschließend wird mit Hilfe asymmetrischer Verschlüsselung oder des Diffie-Hellman-Schlüsselaustauschs ein gemeinsamer symmetrischer Sitzungsschlüssel ausgetauscht um Nutzdaten zu verschlüsseln.
- Eine Reihe von Root-Zertifikaten werden von Browserherstellern akzeptiert und bei der Installation eingetragen.
- Webseiten, die entsprechende Zertifikate haben, werden dann, ebenso wie davon abgeleitete Unter-Zertifikate, bei Aufruf ohne Nachfrage akzeptiert.

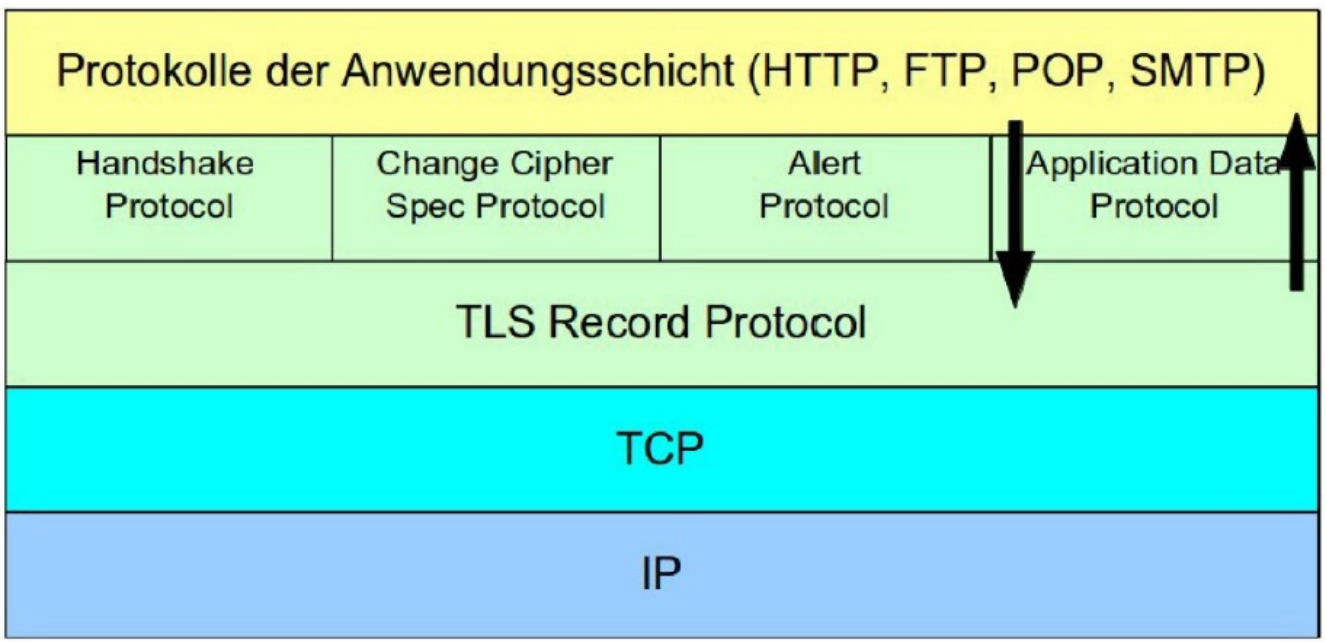
# TLS-Verbindung und TLS-Sitzung

- *TLS-Verbindung:* Eine TLS-Verbindung wird (wie im OSI Referenzmodell) als Transportweg zwischen zwei Endpunkten verstanden. Dabei wird jede Verbindung mit einer Sitzung assoziiert.

- *TLS-Sitzung:* Eine TLS-Sitzung ist eine Assoziation zwischen einem Client und einem Server und wird durch den TLS-Handshake initiiert. TLS-Sitzungen definieren außerdem die kryptographischen Parameter, die für die sichere Datenübertragung nötig sind.

# TLS Protokollstack

- TLS besteht aus mehreren Teilprotokollen, die als TLS-Protokollstack bezeichnet werden
- Insgesamt gibt es fünf TLS-Teilprotokolle, die auf zwei TLS-Schichten angesiedelt sind



# TLS-Layer 1 (TLS Record Protocol)

- Das TLS Record Protocol setzt auf TCP und damit auf die Transportschicht des OSI-Schichtenmodells auf
- Es ist das einzige TLS-Teilprotokoll auf der unteren TLS-Schicht, dem TLS-Layer1
- Das TLS Record Protocol stellt die operativen Dienste von TLS bereit
- Auf Senderseite nimmt es die Daten der oberen Schicht entgegen, teilt sie in Datenstrukturen passender Größe und wendet darauf die ausgehandelten Sicherheitsmaßnahmen wie Verschlüsselung und Message Authentication Codes an
- Das Ergebnis der Verarbeitung heißt TLS Record; Die TLS Records werden an die TCP-Schicht übergeben

# TLS-Layer 2

Auf TLS-Layer2 gibt es insgesamt vier Protokolle:

1. *Handshake Protocol:* dient dem Verbindungsaufbau zwischen Client und Server. Es führt insbesondere die Authentifikation durch und handelt die kryptographischen Verfahren sowie Schlüssel aus.
2. *Change Cipher Spec Protocol:* signalisiert, auf die gerade im Rahmen des Handshake Protocols ausgehandelten Sicherheitsparameter zu wechseln.
3. *Alert Protocol:* ist zuständig für die Behandlung von Fehlern, insbesondere im Rahmen des Handshakes.
4. *Application Data Protocol:* leitet einfach die Daten zwischen Anwendungsschicht und TLS-Layer1 durch (siehe Abbildung)

# Schutzziele und Mechanismen

- TLS soll je nach Wunsch der beiden Kommunikationspartner die folgenden Schutzziele erreichen:
  1. Vertraulichkeit,
  2. Instanzauthentiziät,
  3. Datenauthentizität,
  4. Datenintegrität.
- Als potenzielle kryptographische Verfahren stehen zur Verfügung:
  1. symmetrische Verschlüsselung (Vertraulichkeit),
  2. asymmetrische Public Key Verfahren (Instanzauthentizität)
  3. MACs auf Basis von Hashfunktionen (Datenauthentizität und -integrität)

# Sicherheitsparadigmen

- TLS realisiert zwei wichtige Sicherheitsparadigmen:

1. Verwende einen kryptographischen Schlüssel nur für einen dezidierten Zweck
2. Tausche möglichst wenig Informationen zu geheimen kryptographischen Schlüsseln über das nicht vertrauenswürdige Internet aus

# Einmalschlüssel

- Das erste Paradigma setzt TLS dadurch um, dass für jedes unidirektionale Sicherheitsziel je ein kryptographischer Schlüssel genutzt wird
- Konkret benötigen wir also für TLS mindestens 4 symmetrische Schlüssel: zunächst 2 Schlüssel für den Client als Sender (zum Verschlüsseln und für den MAC) und 2 für den Client als Empfänger (d.h. für den Server als Sender)
- Die Empfängerschlüssel des Clients sind die Senderschlüssels des Servers und umgekehrt (tatsächlich benutzt TLS je 3 Senderschlüssel pro Seite, insgesamt also 6)
- Hintergrund ist, dass für bestimmte Verschlüsselungsmodi ein Initialisierungsvektor benötigt wird

# Austausch kryptographischer Schlüssel

- Um das zweite Paradigma zu berücksichtigen, werden diese Schlüssel aber nie über ein unsicheres Medium übermittelt
- Stattdessen tauschen Client und Server via TLS nur eine einzige Datenstruktur aus: das *Pre-Master-Secret* (PMS)
- Das PMS ist eine Basisinformation zwischen Client und Server, mit der die beteiligten Partner dann dezentral zunächst das gemeinsame Master Secret (MS) ableiten
- Aus dem MS werden ihre symmetrischen Sender- bzw. Empfänger-Schlüssel abgeleitet

# CipherSuite I

- Die Informationen zu den gewünschten Kombinationen kryptographischer Verfahren aus einem asymmetrischen Algorithmus zum Schlüsselaustausch, der symmetrischen Verschlüsselung und einer Hashfunktion wird bei TLS mittels einer *CipherSuite* dargestellt
- Der Client schlägt beim Verbindungsaufbau eine Reihe von CipherSuites vor, der Server wählt daraus eine CipherSuite aus, die zur Absicherung der Client-Server-Verbindung genutzt wird
- Die CipherSuites in SSL/TLS werden nach einem bestimmtem Muster angegeben
- Muster: `TLS_<KeyExchange>_WITH_<Cipher>_<Mac>`

# CipherSuite II

1. *KeyExchange:* das Verfahren zum Austausch des Pre-Master-Secrets (PMS) (d.h. für den Schlüsselaustausch) sowie das asymmetrische Verfahren zur Instanzauthentifikation (je nach Verfahren ist dazu die Angabe eines oder zweier asymmetrischer Verfahren notwendig)
2. *Cipher:* Symmetrisches Verschlüsselungsverfahren zur Verschlüsselung der TLS Records. Ist die Schlüssellänge nicht durch das Verfahren festgelegt, wird sie hier noch angegeben. Sofern die Chiffre eine Blockchiffre ist, wird zusätzlich der Betriebsmodus (oft CBC, GCM) angegeben
3. *Mac:* Hashverfahren zur Berechnung des MACs zur Datenauthentizität und -integrität der TLS Records

# CipherSuite III

- Einige standardisierte CipherSuites aus dem Standard TLS 1.2:

| Cipher Suite | Key Exchange | Cipher | Mac |
|---|---|---|---|
| TLS_NULL_WITH_NULL_NULL | NULL | NULL | NULL |
| TLS_RSA_WITH_RC4_128_MD5 | RSA | RC4_128 | MD5 |
| TLS_RSA_WITH_RC4_128_SHA | RSA | RC4_128 | SHA |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA | RSA | 3DES_EDE_CBC | SHA |
| TLS_RSA_WITH_AES_128_CBC_SHA | RSA | AES_128_CBC | SHA |
| TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 | DHE_RSA | AES_128_CBC | SHA256 |

- Das erste Beispiel nutzt keine Sicherheitsmechanismen!

# Kryptographische Schlüssel, Pre-Master-Secret, Master-Secret I

- Dient RSA zum Schlüsselaustausch, so bestimmt der Client alleine das Pre-Master-Secret (PMS)
- Es ist in diesem Fall eine 48-Byte lange Datenstruktur: 46-Byte stammen von einer Zufallszahl, die verbleibenden zwei Byte sind die TLS Versionsnummer.
- Der Client übermittelt das PMS an den Server, indem er es mit dem RSA-Public-Key des Servers verschlüsselt
- Dann kann nur der Server daraus das PMS mit seinem zugehörigen Private Key berechnen

# Kryptographische Schlüssel, Pre-Master-Secret, Master-Secret II

- Im Fall von Diffie-Hellman als Schlüsselaustauschverfahren erzeugen beide Seiten zunächst ein einmaliges Diffie-Hellman-Schlüsselpaar
- Aus diesem wird nach dem klassischen Diffie-Hellman-Verfahren das PMS abgeleitet
- Beide Seiten kennen also das PMS, obwohl es nie im Klartext übermittelt wurde

# Kryptographische Schlüssel, Pre-Master-Secret, Master-Secret III

- Aus dem Pre-Master-Secret berechnen der Client und der Server dezentral mit Hilfe von Hashfunktionen das 48-Byte lange Master-Secret (MS)
- Daraus berechnen beide Kommunikationspartner abschließend die sechs kryptographischen Schlüssel:

1. Ein symmetrischer Schlüssel $K_C$ für die Verschlüsselung der Daten, die der Client an den Server sendet. In der TLS-Notation wird er mit `client_write_key` bezeichnet
2. Ein symmetrischer Schlüssel $K_S$ für die Verschlüsselung der Daten vom Server. TLS bezeichnet diesen Schlüssel als `server_write_key`

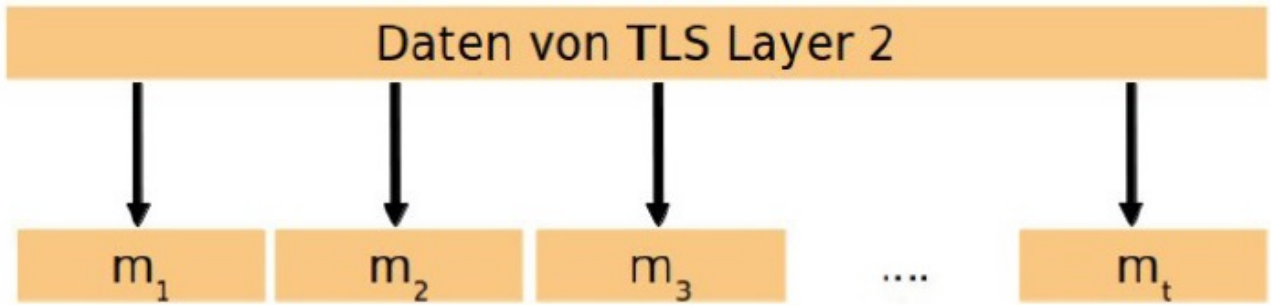
# Kryptographische Schlüssel, Pre-Master-Secret, Master-Secret IV

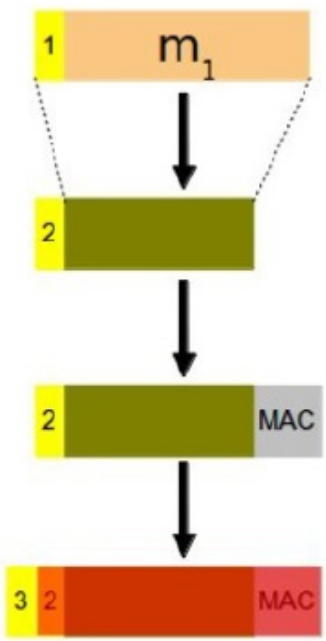
3. Ein symmetrischerMAC-Schlüssel $K_{MAC-C}$ zur Integritätssicherung der TLS Records, die der Client an den Server schickt. TLS nennt diesen Schlüssel `client_write_MAC_key`
4. Ein symmetrischer MAC-Schlüssel $K_{MAC-S}$ zur Integritätssicherung der Daten, die der Client vom Server empfängt. Die TLS-Notation bezeichnet diesen Schlüssel als `server_write_MAC_key`
5. Im Falle einer symmetrischen Blockchiffre gibt es noch die beiden Initialisierungsvektoren `client_write_IV` sowie `server_write_IV`

# TLS Record Protocol I

- Das Record Protocol fragmentiert die Daten des TLS-Layer2 in Fragmente $m_1$, $m_2$, ... von höchstens $2^{14}$ Byte, also 16 KiB
- Jedes Fragment erhält einen Header1, aus dem das TLS-Layer2-Protokoll sowie die TLS-Version hervorgehen
- Dieses Fragment samt Header wird komprimiert, sofern das im TLS-Handshake festgelegt wurde (oft wird keine Komprimierung eingesetzt)
- Das komprimierte Fragment erhält einen neuen Header2 mit den gleichen Informationen wie Header1
- Anschließend wird der MAC über Header2 und komprimiertes Fragment berechnet, danach wird das gesamte Fragment samt Header2 verschlüsselt

# TLS Record Protocol II




Fragmentierung der Daten
Hinzufügen des Headers 1

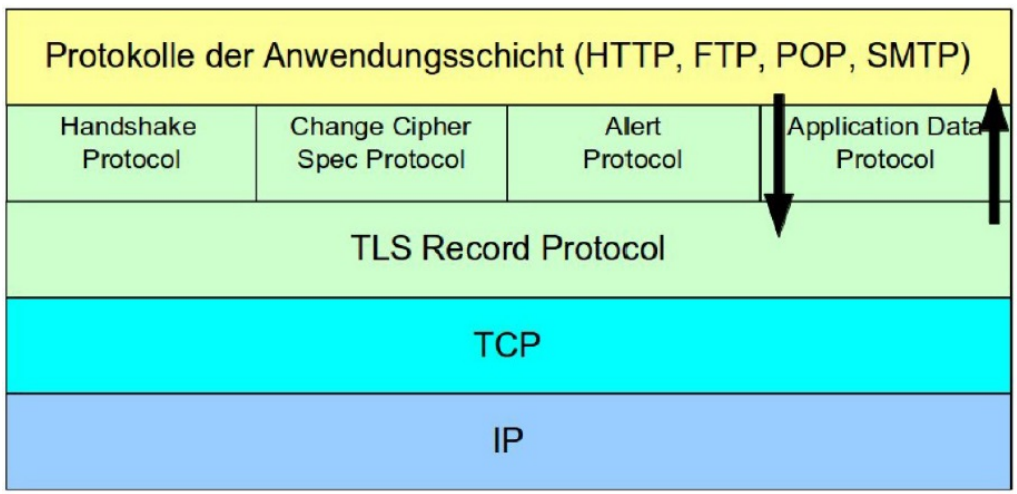
Kompression inklusive Header 1
Hinzufügen des Headers 2

Berechnung u. Anhängen des MAC

Verschlüsselung
Hinzufügen des Headers 3

# TLS Application Data Protocol

- Das Application Data Protocol hat eine einfache Aufgabe, nämlich die Durchleitung der Daten zwischen Anwendungsschicht und TLS-Layer1
- Das Application Data Protocol stellt also die operative Schnittstelle zur Anwendungsschicht dar



# TLS Handshaking Protokolle

- Die übrigen drei Protokolle auf TLS-Layer2 heißen TLS Handshaking Protokolle
  1. TLS Handshake Protocol
  2. TLS Change Cipher Spec Protocol
  3. TLS Alert Protocol
- Diese sind im Rahmen des TLS Handshakes relevant, sie haben keine Schnittstelle zur Anwendungsschicht
- Hinweis: das TLS Handshake Protocol eines der drei TLS Handshaking Protokolle ist
  (Begrifflichkeiten nicht verwechseln!)

# TLS Handshake Protocol

- Das TLS Handshake Protocol dient dem Verbindungsaufbau zwischen Client und Server
- Im Rahmen des TLS Handshakes findet die Authentifikation des oder der Kommunikationspartner, das Aushandeln der zu verwendenden kryptographischen Verfahren und der Austausch benötigter geheimer Informationen statt
- Es gibt für die Authentifikation drei Möglichkeiten: keine Authentifikation, nur der Server authentisiert sich oder beide authentisieren sich
- Wenn der TLS Handshake abgeschlossen ist, liegen die kryptographischen Verfahren fest und beide Seiten haben Zugriff auf alle sechs Sitzungsschlüssel

# TLS Change Cipher Spec Protocol

- Das Change Cipher Spec Protocol umfasst lediglich eine Nachricht bestehend aus dem Klartext-Byte mit dem Wert 1
- Sie wird im Rahmen des TLS Handshakes gesendet
- Mit dieser Nachricht signalisiert der Sender, dass er für die folgenden TLS-Records auf die gerade festgelegten Verfahren und Schlüssel umsteigt

# TLS Alert Protocol

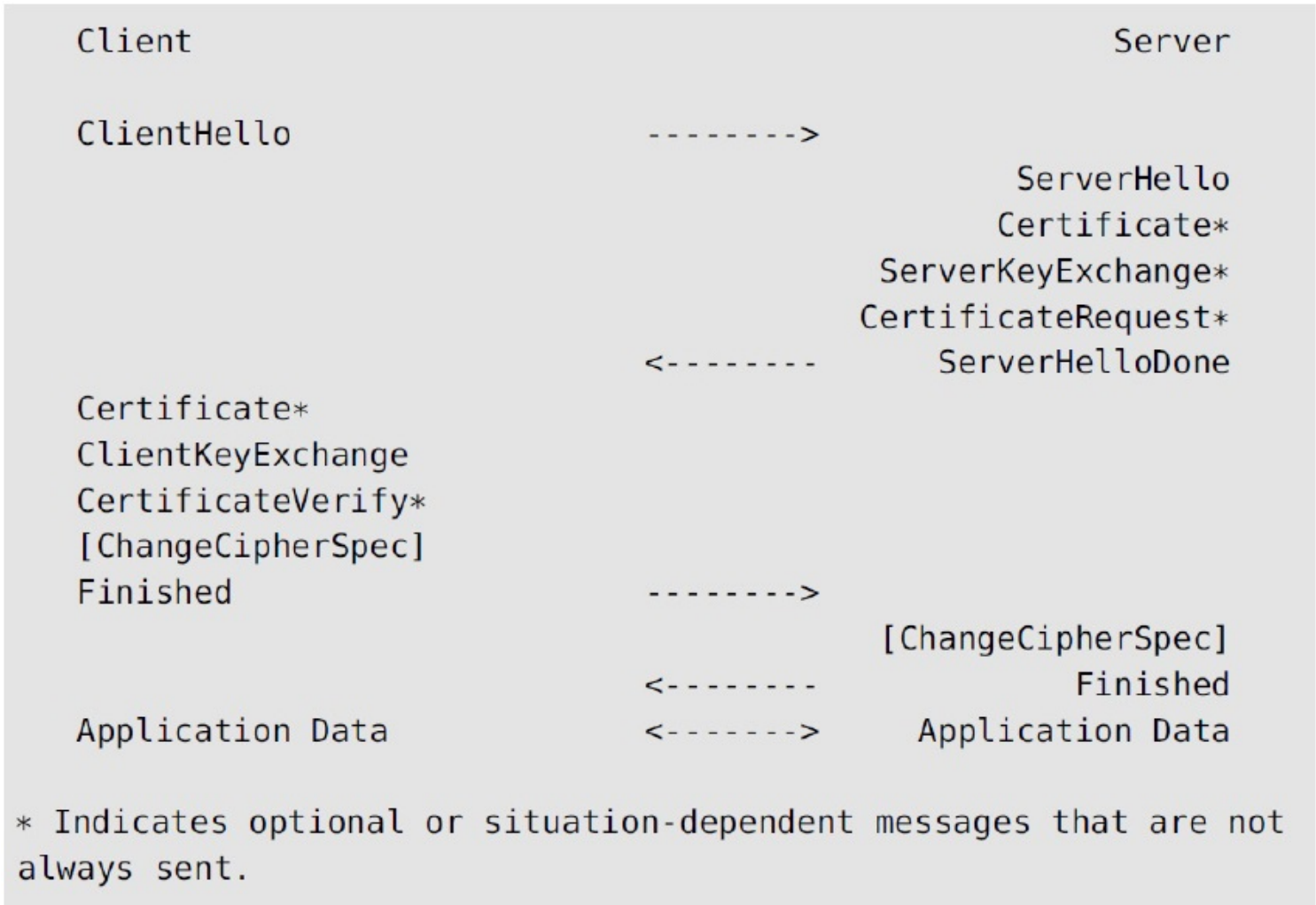
- Mit diesem Protokoll werden TLS-spezifische Warnungen an den Kommunikationspartner übermittelt
- Eine Alert-Nachricht besteht aus zwei Bytes
- Mit dem ersten Byte wird die Schwere der Warnmeldung angezeigt (Warning $= 1$, Fatal $= 2$)
- Wird ein fataler Zustand signalisiert, führt das zum sofortigen Abbruch der Verbindung
- Ebenso werden keine neuen Verbindungen für diese Sitzung mehr eröffnet
- Das zweite Byte kodiert Hinweise zum Fehler, zB: `decompression_failure`, `protocol_version`, `decrypt_error`, usw.

# TLS-Handshake I

- Der Handshake Aufbau einer durch TLS gesicherten Verbindung wird auch als TLS-Handshake bezeichnet.
- Die wesentlichen Ziele des TLS Handshakes sind:

1. Festlegung der verwendeten kryptographischen Verfahren für die Absicherung der TLS-Records
2. Festlegung der Komprimierung bzw. ob komprimiert wird
3. Festlegung, wer sich authentisiert sowie Durchführung der Authentifikation durch den Kommunikationspartner (in den meisten Fällen authentisiert sich nur der Server mittels TLS)

# TLS-Handshake II



# TLS-Handshake III

- `ClientHello`: Zunächst signalisiert der Client dem Server, dass er mit ihm eine TLS-Sitzung aufbauen möchte
- Dazu sendet der Client eine ClientHello-Nachricht
- Darin teilt der Client die von ihm unterstützten CipherSuites mit
- Außerdem schickt er zur Vermeidung von Replay-Angriffen eine ID sowie eine vom Client gewählte Zufallszahl RND1 an den Server

# TLS-Handshake IV

- `ServerHello`: Der Server antwortet mit einer ServerHello-Nachricht antwortet
- Darin teilt der Server die von ihm festgelegte CipherSuite für diese Sitzung mit
- Außerdem schickt er die Client-ID sowie seine Zufallszahl RND2 zurück
- Anzumerken ist, dass der Client lediglich eine Liste der unterstützten Verfahren sendet und der Server über das Verfahren entscheidet

# TLS-Handshake V

- `(Server)Certificate`: Soll der Server authentifiziert werden, sendet er anschließend mit der Certificate-Nachricht sein Zertifikat
- Soll der Server sich nicht authentisieren, unterbleibt die Certificate-Nachricht
- Daher ist sie als optional markiert

# TLS-Handshake VI

- `ServerKeyExchange`: Des Weiteren wird vom Server je nach festgelegter CipherSuite eine ServerKeyExchange-Nachricht gesendet
- Beispielsweise wenn Diffie-Hellman als Schlüsselaustauschmethode verwendet wird
- Im Fall einer CipherSuite der Form `TLS_RSA_WITH...` sendet der Server keine ServerKeyExchange-Nachricht, weil der Client das Pre-Master-Secret wählt und RSA-verschlüsselt an den Server schickt

# TLS-Handshake VII

- `CerficateRequest`: Optional verlangt der Server eine TLS-Client-Authentifikation
- dazu sendet er eine CerficateRequest-Nachricht.
- `ServerHelloDone`: Zum Abschluss sendet der Server eine ServerHelloDone-Nachricht, um dem Client zu signalisieren, dass der Server auf die Client-seitigen Nachrichten wartet
- `(Client)Certificate`: Sofern der Server eine TLS-Client-Authentisierung wünscht, sendet der Client mittels einer Certificate-Nachricht sein Zertifikat an den Server
- Andernfalls entfällt die (Client)Certificate-Nachricht

# TLS-Handshake VIII

- `ClientKeyExchange`: In jedem Fall sendet der Client eine ClientKeyExchange-Nachricht
- Im Fall von Diffie-Hellman als Schlüsselaustauschverfahren sendet der Client seinen DH-Public-Key an den Server
- Im Fall von RSA wählt er das PMS, verschlüsselt es mit dem RSA-Public-Key des Servers und sendet es als ClientKeyExchange-Nachricht
- `CertificateVerify`: Im Falle einer TLS-Client-Authentisierung signiert der Client mit dem zu seinem Zertifikat gehörenden Private Key alle bisherigen Handshake-Nachrichten
- Er sendet diese Signatur als CertificateVerify-Nachricht zum Server

# TLS-Handshake IX

- `ChangeCipherSpec`: Der letzte Schritt des Clients beginnt mit der ChangeCipherSpec-Nachricht, die angibt, dass der Client fortan seine gesendeten Nachrichten mit den ausgehandelten kryptographischen Verfahren und Schlüsseln absichert
- `Finished`: Direkt darauffolgend wird eine Finished-Nachricht gesendet, die einen Hashwert enthält, der über alle empfangenen und gesendeten Nachrichten gebildet wird und mit den neuen Sicherheitseinstellungen abgesichert wird
- Dadurch zeigt der Client, dass er das PMS kennt

# TLS-Handshake X

- Der TLS-Handshake wird dadurch abgeschlossen, dass auch der Server den Umstieg der von ihm gesendeten TLS-Records auf die neuen Sicherheitseinstellungen mittels einer ChangeCipherSpec-Nachricht signalisiert
- Danach weist er durch eine gültig abgesicherte Finished-Nachricht nach, dass auch er das PMS kennt, weil er die daraus abgeleiteten Schlüssel nutzt
- Ab diesen Zeitpunkt werden alle Informationen mit den neuen Sicherheitseinstellungen abgesichert

# Sicherheit von TLS I

- Eine wichtige Beobachtung ist, dass TLS selbst als sicher zu betrachten ist
- In der Praxis ist oft der Mensch zu einer Fehlhandlung 'motiviert' (Social Engineering) oder eine Implementierung zeigt sich als fehlerhaft
- *Authentifikation:* wichtiger Punkt zur Sicherheit von TLS ist die Art der Authentifikation
- Zunächst bestimmt allein der Server die CipherSuite und damit das Authentifikationsverfahren
- Der Client muss also seine Vorschlagsliste auf CipherSuites beschränken, die er für sicher hält

# Sicherheit von TLS II

- Weiterhin ist die Prüfung der Zertifikatskette des Public Key des Servers sicherheitskritisch; diese übermittelt der Server mit seiner Certificate-Nachricht
- Die Zertifikatskette muss aus vertrauenswürdigen Zertifikaten bestehen, insbesondere mit einem solchen enden
- Andernfalls sind Phishing-Angriffe auch über eine HTTPS-Verbindung möglich

# Sicherheit von TLS III

Verbindlichkeit und Pseudonymität/Anonymität:

- Die übertragenen Daten werden nicht signiert, TLS erzielt also keine Verbindlichkeit von Aktionen (meist nicht weiter wichtig)
- Relevanter ist, dass TLS keine Maßnahmen zur Abwehr von Verkehrsflussanalysen bereitstellt, da nur die Nutzdaten in den TCP/IP-Paketen verschlüsselt werden
- Das Sicherheitsziel Pseudonymität oder gar Anonymität ist außerhalb des Fokus von TLS

# Sicherheit von TLS IV

CipherSuite:

- Die Sicherheit des Protokolls hängt auch von den verwendeten kryptografischen Verfahren ab, die die Kommunikationspartner im Handshake miteinander abstimmen
- Falls ein Angreifer dafür sorgen kann, dass die Kommunikationspartner schwache Verschlüsselungsverfahren oder schwache Schlüssel aushandeln, könnte er anschließend versuchen, den verwendeten Kommunikationsschlüssel zu brechen

# Sicherheit von TLS V

Forward Secrecy:

- Eine wichtige Eigenschaft ist Forward Secrecy
- Das bedeutet, dass ein in der Vergangenheit ausgetauschtes PMS weiterhin sicher bleibt, selbst wenn ein Private Key (typischerweise der des TLS-Servers) heute kompromittiert wird und die alte TLS-Kommunikation gespeichert wurde
- Die CipherSuite `TLS_RSA_WITH_...` gewährleistet kein Forward Secrecy, weil das PMS mit dem kompromittierten RSA-Private Key berechnet werden kann
- Daher verwenden heutige TLS-Verbindungen CipherSuites der Form `TLS_DHE_RSA_WITH_...`, da die Diffie-Hellman-Schlüssel genau einmal verwendet werden

# Sicherheit von TLS VI

TLS 'gebrochen':

- Des Öfteren lesen Sie über Angriffe, die SSL/TLS 'gebrochen' haben sollen
- Meist wird nicht das TLS-Konzept selber kompromittiert, sondern der Angreifer hat spezielle Voraussetzungen auf dem Zielrechner oder es betrifft eine bestimmte Implementierung
- Beispiele:
  - BEAST: Angreifer hat Zugriff auf ein Java Applet,
  - CRIME: hier muss Komprimierung eingesetzt werden,
  - Poodle-Angriff: zielt auf die Rückwärtskompatibilität von TLS ab
  - Heartbleed (nächste Folie)

# Sicherheit von TLS VII

Heartbleed I:

- Der Heartbleed-Angriff nutzt eine Schwachstelle in der Heartbeat-Erweiterung der verbreiteten TLS-Implementierung `openssl` aus
- Über eine Heartbeat-Anfrage kann der Client den Server fragen, ob dieser noch verfügbar ist
- Dazu wird eine Nachricht gesendet, die der Server wiederholen soll (echo-Nachricht)

# Sicherheit von TLS VIII

Heartbleed II:

- Die Schwachstelle besteht darin, dass der Client mehr Zeichen vom Server zurückfordert als er sendet
- In diesem Fall liest der Server zum Auffüllen seiner echo-Antwort benachbarte Inhalte seines Hauptspeichers aus und sendet diese an den Client
- In dem betroffenen RAM-Bereich des Servers können aber sensible Daten (z.B. private Schlüssel) gespeichert werden

# Ausblick: IPsec I

- Beim Entwurf von IP wurden keine Sicherheitsmechanismen integriert
- Das Internet-Protokoll-Security (IPSec) ist ein Rahmenwerk, welches in einem IP-Netz folgende Schutzziele erfüllt:
  1. Vertraulichkeit,
  2. Authentizität,
  3. Integrität
- Dazu werden verschiedene Mechanismen eingesetzt, etwa Verschlüsselung einzelner IP-Pakete und Einfügen eines zusätzlichen Paket-Headers mit einem MAC

# Ausblick: IPsec II

- IPSec wird in einer Reihe von Standarddokumenten, den Request For Comments (RFCs) definiert
- Beim Einsatz von IPSec kann entschieden werden die Daten entweder nur zu Verschlüsseln, nur zu Authentifizieren, oder sie zu Verschlüsseln und zu Authentifizieren
- Die IPSec-Spezifikation umfasst dabei drei Protokolle:
  1. Das Internet Key Exchange (IKE) für die Autorisierung der Kommunikationspartner und deren Austausch von Schlüsseln bzw. Schlüsselparametern
  2. Das Encapsulating Security Payload (ESP) für den verschlüsselten und integren Datentransfer
  3. Den Authentication Header (AH) für authentifizierten Datenaustausch

# Ausblick: IPsec III

- Für die Verwendung von AH und ESP gibt es zwei verschiedene Modi:

1. Transport-Modus: Sicherung der Nutzdaten
2. Tunnel-Modus: Schutz des gesamten IP-Paketes; dazu wird das zu schützende IP-Paket in ein neues IP-Paket verpackt

# Ausblick: IPsec IV

- Das IKE-Protokoll arbeitet dabei in zwei Phasen
- In der ersten Phase wird eine Verbindung mit Sicherheitsparameter ausgehandelt
- Die Verbindungen von IPSec Security Association werden als Sicherheitsassoziationen (engl.: Security Association (SA)) bezeichnet
- Jeder Kommunikationspartner speichert zu seiner SA alle Daten, die für die kryptographische Verarbeitung der zugehörigen Daten notwendig sind
- Die Datenstruktur, in der SAs gespeichert werden, heißt Security Association Database (SAD)

# Ausblick: IPsec V

- Diese Verbindungen gelten jeweils nur in eine Richtung, sodass für eine bidirektionale Kommunikation von zwei Parteien auch zwei SAs nötig sind
- Eine SA zwischen den Kommunikationspartnern beinhaltet die,

1. Identifikation der Kommunikationspartner z.B. mit IP-Adressen oder Zertifikaten
2. Festlegung der eingesetzten Kryptoalgorithmen
3. Quell- und Zieladresse im (IP)-Netz für die IPSec-Verbindung
4. Zeitspanne, in der eine erneute Authentifizierung erforderlich wird und in der IPSec-Schlüssel zu erneuern sind

# Ausblick: IPsec VI

- In Phase 1 werden Parameter, wie die Lebensdauer und die Authentisierungsmethoden für eine IPSec-Verbindung ausgehandelt
- Anschließend werden in Phase 2 entweder Zertifikate oder Schlüssel aus einem zuvor vereinbartem Geheimnis, so Pre-Shared-Keys genannte Pre-Shared Keys (PSK) verwendet um die Autorisierung umzusetzen
- Verwendungszweck: IPSec wird hauptsächlich zur Realisierung von sogenannten Virtual Private Networks (VPNs) benutzt, welche oft in Firmenumgebungen eingesetzt werden