Lösung 1 (Allgemeine Grundlagen) **(6+3+6) = 15 Punkte**

a) Erläutern Sie knapp die Begriffe *Dienst*, *Protokoll* und *Dienstprimitiv*. Machen Sie in Ihrer Erläuterung auch klar, wie diese Begriffe in Zusammenhang stehen.

Vom Netz bzw. einer der Schichten bereitgestellte Kommunikationsfunktionen werden als Dienst bezeichnet. Im Prinzip wird eine Schnittstelle zur Kommunikation definiert, die das 'was' einer Kommunkationssoftware beschreibt.

Die Dienste werden durch bestimmte Softwareinstanzen erbracht, welche sich nach bestimmten Regeln verhalten und mit anderen Instanzen (auf anderen Rechnern) interagieren – diese Instanzen, die die Regeln implementieren, sind die Ptrotokolle (das 'wie' der Dienste).

Ein Dienst ist eine Sammlung von einzelnen Funktionen, die genutzt werden können – diese Funktionen bauen auf Dienstprimitiven auf, welche als Basisbausteine die Interaktion der Dienstinstanzen ermöglichen. (Je nach aufgerufenem Dienstprimitiv wird gemäß einer bestimmten Regel agiert und eventuell ein anderes Dienstprimitiv ausgelöst.)

Also: Dienste beschreiben die Funktionalität, Protokolle implementieren sie und Dienstprimitive sind die Basisbausteine der Dienste.

b) Erläutern Sie knapp, was man unter *vertikaler* und *horizontaler* Kommunikation versteht. Verwenden Sie für die Erläuterung ein *Beispiel aus dem Internet-Referenzmodell*.

Horizontale Kommunikation: Zwischen (verteilten) Instanzen der gleichen Schicht, zwischen verschiedenen Kommunikationspartnern, außer auf unterster Ebene nur scheinbare Kommunikation.

Ein Beispiel dafür ist das Senden eines Pakets durch IP; das Paket ist scheinbar direkt an die empfangende IP-Instanz gerichtet.

Vertikal: Tatsächlich erbringen die tieferen Schichten diesen Kommunikationsdienst, d.h. tatsächliche Kommunikation verläuft innerhalb des Protokollstapels jedes Kommunikationspartners.

Ein Beispiel dafür ist die Übergabe eines IP-Pakets an die Netzwerkkarte, um die Zustellung des Pakets zum nächsten Hop zu ermöglichen.

(Verschiedenste Beispiele möglich)

c) Sie verwenden eine Leitung, die auf der Bitübertragungsschicht eine Übertragungsrate von 1GBit/s bietet. Welche Ursachen können dazu führen, dass eine Anwendung, die auf dieser Leitung Daten überträgt, eine geringere Übertragungsrate misst? *Nennen Sie drei unterschiedliche Ursachen und erläutern Sie jeweils knapp, wieso sie zu einer geringeren Übertragungsrate führen.*

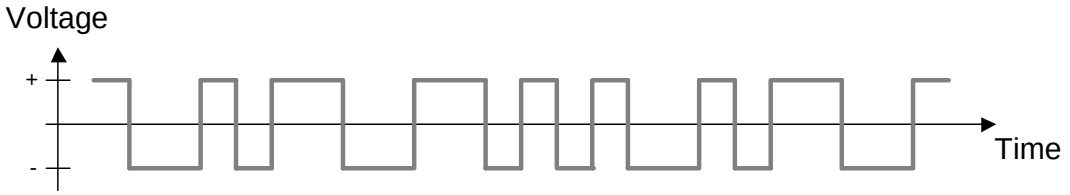
Wichtige Ursachen zur geringeren Übertragungsrate einer TCP-basierten Anwendung (man kann natürlich auch UDP nehmen, aber dann hat man eine geringere Menge an Ursachen):

- TCP-Flusskontrolle: zu wenig Buffer (zu kleines Fenster bei Sliding Window) oder zu geringe Verarbeitungsgeschwindigkeit beim Empfänger). Dadurch bekommt der Sender nicht ausreichend schnell Quitungen, um durchgehend senden zu können.
- Protokoll-Overhead: die Übertragungsrate ist inklusive aller Protokollheader, netto bleibt für die Anwendung weniger über.
- Paketverlust: es entsteht eine Verzögerung durch Neuübertragung. (Entweder bei TCP oder auf Layer 2, aber falls beides genannt wird wird nur eins gewertet.) Die Anwendung kann den für Neuübertragungen genutzten Anteil an der Datenrate nicht selbst nutzen.
- Medienzugriff: falls eine geteilte Leitung vorliegt, über die auch andere übertragen können, werden wir eventuell blockiert, siehe CSMA/CD. Dadurch entstehen Pausen im Sendeprozess der Anwendung.
- Congestion Control: TCP drosselt Senderate, weil Router auf IP-Ebene überlastet sind und Pakete verwerfen.
- Multiplexing: andere Anwendungen auf dem gleichen Rechner nutzen die Leitung gleichzeitig, so dass für unsere Anwendung weniger übrig bleibt.
- ...

Nicht korrekt ist es zu sagen, dass der Manchester-Code verwendet wird und daher nur die halbe Datenrate zur Verfügung steht – denn es ist ja gesagt, dass 1 GBit/s geboten werden. Für andere Aspekte der Bitübertragungsschciht gilt dies genauso.

Lösung 2 (Signale) **(4+2+2+4) = 12 Punkte**

a) Welche *Bitfolge* erhält der Empfänger, wenn er folgendes Signal mit dem *Manchester-Code* decodiert?



Es sei nun eine Schrittgeschwindigkeit von `10.000 baud` gegeben. Welche *Datenrate* wird bei Verwendung des Manchester-Codes erreicht?

Bitfolge: 100101110010 (oder das Inverse, je nachdem, welche Variante man verwendet)

Datenrate: 5000 Bit/s (es werden zwei Schritte pro Bit benötigt)

b) Gegeben sei ein neuer Code mit dem Namen *9B/10B*. Welche *Effizienz* kann dieser Code wohl erreichen? Begründen Sie ihre Antwort.

90 Prozent – 9 Bit werden auf 10 Bit codiert, analog zum Prinzip des 4B/5B-Codes.

c) Nennen Sie einen *Vorteil* und einen *Nachteil* des Manchester-Codes gegenüber dem NRZ-L-Code.

Vorteil: Selbsttaktung (oder Gleichstromfreiheit)

Nachteil: Effizienz

d) Gegeben sei ein Kanal mit einer Bandbreite von 3.000 Hz. Der Signal-Rauschabstand beträgt 30 dB. Mit Hilfe des Shannon-Theorems haben Sie bereits berechnet, dass die maximale Datenrate 30.000 Bit/s beträgt.
Nun entschließen Sie sich, den NRZ-L-Code einzuseten. *Berechnen Sie die maximale Datenrate mit Hilfe des Nyquist-Theorems*. Falls Sie dabei auf einen anderen Wert kommen als durch das Shannon-Theorem vorgegeben: *welcher der beiden Werte ist für Sie maßgebend und warum*?

Nyquist-Theorem: $2 \cdot B \cdot ld(n)$ mit B: Bandbreite und n: Signalstufen.

Hier: $B = 3000Hz$ und $n = 2$. Damit ergeben sich 6.000 Bit/s.

Beide Theoreme berücksichtigen unterschiedliche Vorgaben (SNR vs. Signalstufen): Shannon gibt an, welche Datenrate bei der aktzuellen SNR maximal erreicht werden könnte, aber es hängt von der Wahl der Anzahl der Signalstufen ab, ob man sie auch tatsächlich erreichen kann. Nyquist gibt nur an, welche Datenrate durch die Zahl der Signalstufen erreicht werden kann, aber ob diese auch beim Empfänger sauber getrennt werden können, hängt von der SNR ab. Man muss also immer beides berücksichtigen, daher ist das Minimum beider Werte maßgebend – hier also die 6.000 Bit/s.

Lösung 3 (Sicherungsschicht) **(4+5+6+6) = 21 Punkte**

a) Schicht 2 des OSI-Referenzmodells ist in zwei Teilschichten aufgeteilt, die Logical Link Control (LLC) und die Medium Access Control (MAC). *Warum macht man diese Aufteilung und welche Aufgaben haben die beiden Teilschichten jeweils*?

Aufgaben LLC: Flusssteuerung

Aufgabe MAC: Fehlerbehandung (fehlererkennender oder -korrigierender Code), Medienzugriffskontrolle für geteilte Medien.
Aufteilung: LLC übernimmt alle Aufgaben, die unabhängig vom konkreten Netz sind; MAC ist hardwarebezogen und variiert je nach verwendetem Netz. Durch diese Aufgaben braucht man für ein konkretes Netz nur die MAC-Funktionalität zu erstellen; die LLC-Funktionalität muss nur ein Mal implementiert werden und kann immer wiederverwendet werden.

b) Sie verwenden *Cyclic Redundancy Checksum (CRC)* zur Erkennung von Übertragungsfehlern. Zwei Kommunikationspartner haben sich auf die Verwendung des Generatorpolynoms

$$G(x) = x^4 + x^2 + 1$$

geeinigt. Einer der beiden empfängt die folgende Bitsequenz:

1 0 0 0 1 1 0 1 1 1.

Ist ein Übertragungsfehler aufgetreten? Begünden Sie Ihre Antwort!

```
1 0 0 0 1 1 0 1 1 1 : 1 0 1 0 1
1 0 1 0 1
---------
      1 0 0 1 0
      1 0 1 0 1
      ---------
          1 1 1 1 1
          1 0 1 0 1
          ---------
              1 0 1 0 1
              1 0 1 0 1
              ---------
                      0
```

Grund: Rest = 0 $\Rightarrow$ Vermutlich kein Übertragungsfehler – wäre ein Übertragunsfehler aufgetreten, hätten wir einen Rest ungleich 0, außer wenn der Fehler dafür sorgt, dass die Sequenz immer noch Vielfaches des Generatorpolynoms ist. Solche Fehler können wir nicht entdecken, und so einer könnte auch hier aufgetreten sein.

c) Sie verwenden das *Sliding-Window-Verfahren* zusammen mit *Go-Back-N* zur Fehlerbehandlung. Es seien sowohl positive Quittungen (ACK) als auch negative Quittungen (NAK) möglich. *Beschreiben Sie beide Mechanismen knapp anhand der unten stehenden Fragen zum folgenden Beispiel*:

Gegeben seien ein Modulus $M = 11$ und eine Fenstergröße $W = 8$. Zum aktuellen Zeitpunkt seien die Rahmen mit den Sequenznummern 8, 9, 10, 0, 1 vom Sender gesendet worden, ohne dass eine Quittung eingegangen ist.

- Welche Rahmen dürfen in dieser Situation ohne jede Quittung gesendet werden?
- Wie ändert sich die Situation, falls ein Rahmen mit einer Quittungsnummer (ACK) 10 empfangen wird?
- Was passiert, wenn stattdessen ein NAK für die Sequenznummer 0 empfangen wird?

Die Fenstergröße gibt vor, wie viele Rahmen gleichzeitig unbestätigt unterwegs sein dürfen. Hier wurden bereits fünf versendet, drei dürfen also noch gesendet werden: Rahmen 2, 3 und 4.

Wird eine Quittung empfangen, bestätigt sie alles bis zu der vorherigen Sequenznummer. Im Beispiel ist bis Rahmen 9 alles quittiert, Rahmen 10, 0 und 1 sind noch unbestätigt, daher werden nur zwei weitere Positionen im Fenster frei; zusätzlich zu 2, 3 und 4 dürfen jetzt auch Rahmen 5 und 6 versendet werden.

Eine negative Quittung quittiert alles bis zur vorherigen Sequenznummer und gibt an, dass der aktuelle Rahmen fehlt. Ab diesem muss alles neu übertragen werden, da der Empfänger nichts puffert, was out-of-order ankommt. Im Beispiel: Rahmen 8 - 10 sind bestätigt, ab Rahmen 0 muss alles neu übertragen werden, also zumindest die 0 und die 1 (je nachdem, auf welche vorherige Situation man Bezug nimmt, ob mit oder ohne Übertragung von 2, 3 und 4) – und bis hin zu Rahmen 7 kann jetzt alles gesendet werden.

d) Gegeben sei ein Netzwerk mit einer Bustopologie mit einer Ausdehnung von 50m, in dem CSMA/CD eingesetzt wird. Die Datenrate betrage 1 GBit/s, die Signalgeschwindigkeit im physikalischen Medium sei $2 \cdot 10^8 m/s$.

- Wieviel *Zeit* kann *maximal* vergehen, bis eine sendende Station eine *Kollision* erkennt?
- Welche *minimale Rahmenlänge* wäre für dieses LAN erforderlich?
- Sie wechseln nun zu einer Sterntopologie mit *Switch*, behalten aber Datenrate und Ausdehnung bei. *Ändert sich die erforderliche minimale Rahmenlänge? Begründen Sie Ihre Antwort!*

Zeit im Worst-Case gleich doppelter Signallaufzeit zur Gegenseite:

$t_{max} = \frac{2 \cdot 50m}{2 \cdot 10^8 m/s} = \frac{5 \cdot 10^1}{10^8} s = 5 \cdot 10^{-7} s$

Minimale Rahmenlänge: wieviel kann man in $5 \cdot 10^{-7}$ Sekunden versenden?

$r_{min} = 1GBit/s \cdot 5 \cdot 10^{-7} s = 10^9 \cdot 5 \cdot 10^{-7} GBit = 500Bit$

Bei einem Switch können keine Kollisionen mehr auftreten, drum ist die Festlegung einer minimalen Rahmenlänge unerheblich, es könnten beliebig kleine Rahmen versendet werden.

Lösung 4 (Internet Protocol (IP)) **(6+2+2+4+4) = 18 Punkte**

a) Gegeben sei ein Netz mit dem IP-Adressbereich 137.226.28.0/22. Dieses möchten Sie unter kompletter Ausnutzung des zur Verfügung stehenden Adressraums in 6 Subnetze zerlegen, wobei zwei der Subnetze jeweils doppelt so groß sein sollen wie jedes der 4 anderen. *Geben Sie die Adressbereiche der Subnetze an.*

Subnetzmaske /22 : 10 Bit.

Lösungsansatz: Zerteile das Netz in 4 Subnetze. Davon zerteile zwei wiederum in jeweils zwei; dadurch entstehen zwei große und vier kleine Subnetze.

Lösungsmöglichkeit:

- Großes Netz 1: 137.226.28.0/24
- Großes Netz 2: 137.226.29.0/24
- Kleines Netz 1: 137.226.30.0/25
- Kleines Netz 2: 137.226.30.128/25
- Kleines Netz 3: 137.226.31.0/25
- Kleines Netz 4: 137.226.31.128/25

b) Woran kann ein Sender erkennen, dass die Ziel-IP-Adresse eines IP-Paketes sich *nicht in seinem eigenen Subnetz* befindet?

Der Sender erkennt, dass das sein eigenes Netzpräfix nicht gleich dem des Empfängers ist.
Dazu verwendet er seine Subnetzmaske: eigene IP-Adresse & eigene Subnetzmaske != Ziel-IP-Adresse & eigene Subnetzmaske.

c) Ein Router habe ein IP-Paket erhalten, welches in sein eigenes Subnetz weitergeleitet werden muss. Was ist der *erste Schritt*, den der Router tätigen muss, um das Paket im Subnetz zuzustellen und *welches Protokoll* wird dazu verwendet?

Zunächst muss die MAC-Adresse des Rechners ermittelt werden, dem die entsprechende IP-Adresse zugewiesen ist.

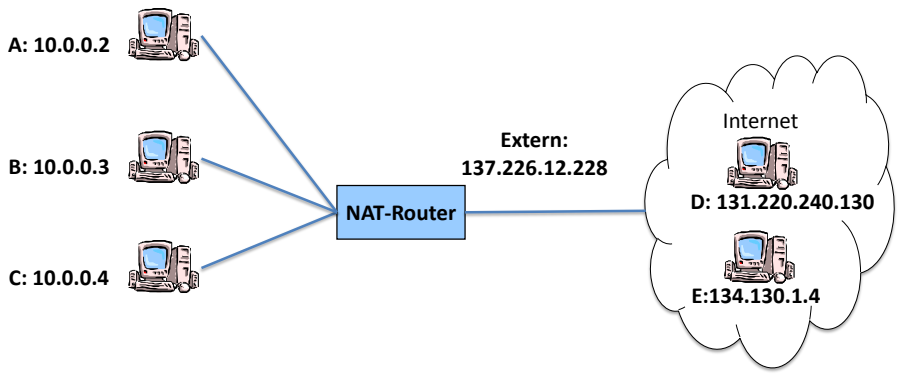
Das dazu benötigte Protokoll ist ARP.

d) Zwei wichtige Felder im IP-Header sind *Protocol* und *TTL*. *Wer prüft jeweils diese Felder und zu welchem Zweck*?

Protocol: wird vom Empfänger geprüft, also dem Rechner, dessen IP-Adresse im Paket als Destination angegeben ist. (Und im Prinzip auch von Middleboxes wie NAT, Firewall, aber das Middlebox-Konzept generell war nciht Thema der Vorlesung.)
Zweck: Ermittlung des Payload-Typs, um die Daten auf dem Zielrechner korrekt weiterverarbeiten zu können.

TTL: wird von jedem Router, der das Paket empfängt, geprüft (und um 1 dekrementiert).
Zweck: Vermeiden der endlosen Weiterleitung von Paketen (bei Router-Fehlkonfiguration).

e) Gegeben ist das folgende kleine Firmennetzwerk mit den drei Rechnern A, B und C. Zur internen Kommunikation werden die angegebenen privaten IP-Adressen verwendet. Zugang zum Internet erfolgt mittels eines NAT-Routers, der nach außen hin die IP-Adresse 137.226.12.228 besitzt.



Die Abbildungstabelle des NAT-Routers sei zunächst leer. Rechner A sendet nun an D mit Absender-Port 134 und Ziel-Port 80. *Welche Aktion führt der NAT-Router durch und welche Einträge legt er dabei in seiner Abbildungstabelle an*? Kurz darauf sende Rechner E an 137.226.12.228:4936. *Welche Aktion führt der NAT-Router nun durch?*

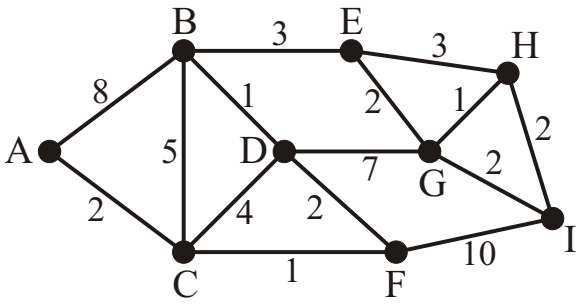
Der NAT-Router ersetzt die nur lokal gültige Absenderadresse durch seine externe Adresse, bevor er das Paket weiterleitet.

Dabei wird der folgende Eintrag angelegt: 10.0.0.2:134 137.226.12.228:134 131.220.240.130:80 (oder in welcher Reihenfolge und Sytax auch immer). Wesentlich hierbei sind die ersten beiden Adresstupel, das dritte wird in der Praxis auch angelegt, ist hier aber nicht Kern der Frage.

Das von E kommende Paket wird einfach verworfen, da kein passender Eintrag zur Weiterleitung existiert.

Lösung 5 (Routing) **(8+4+2) = 14 Punkte**

Gegeben sei das folgende Netzwerk. Die Kanten sind mit der Entfernung der anliegenden Knoten beschriftet.



a) Berechnen Sie mit Hilfe des *Dijkstra-Algorithmus* den kürzesten Pfad von A nach I. Ergänzen Sie hierfür die folgende Tabelle, indem Sie spaltenweise die einzelnen Schritte des Algorithmus' dokumentieren. Verwenden Sie Einträge der Form (n,X), die den Knotenbeschriftungen des jeweiligen Schritts entsprechen. Dabei ist $n \in \mathbb{N}$ die Länge des bisher kürzesten Weges zum betrachteten Knoten und $X \in \{A, \ldots, I\}$ der unmittelbare Vorgängerknoten auf diesem Weg. Wird ein Knoten als permanent markiert, soll dies durch eine zusätzliche Umrahmung der Markierung notiert werden (siehe Knoten A im Schritt 0). Alle noch nicht erreichbaren Knoten werden mit ∞ beschriftet. Um Schreibarbeit zu sparen, brauchen die Markierungen von bereits als permanent markierten Knoten nicht mehr in jeder Spalte wiederholt werden (vgl. Zeile A).

	0	1	2	3	4	5	6	7	8
A	**(0,-)**	-	-	-	-	-	-	-	-
B	∞	**(8,A)**	**(7,C)**	**(7,C)**	**(6,D)**	-	-	-	-
C	∞	**(2,A)**	-	-	-	-	-	-	-
D	∞	∞	**(6,C)**	**(5,F)**	-	-	-	-	-
E	∞	∞	∞	∞	∞	**(9,B)**	-	-	-
F	∞	∞	**(3,C)**	-	-	-	-	-	-
G	∞	∞	∞	∞	**(12,D)**	**(12,D)**	**(11,E)**	-	-
H	∞	∞	∞	∞	∞	∞	**(12,E)**	**(12,EG)**	-
I	∞	∞	∞	**(13,F)**	**(13,F)**	**(13,F)**	**(13,F)**	**(13,FG)**	**(13,FG)**

Kürzester Pfad: ACFI

Achtung, siehe Tabelle: es existiert noch ein weiterer, genauso langer Pfad. Eigentlich wird nur ein Vorgänger gemerkt, keine zwei wie oben eingetragen. Welcher, ist implementierungsabhängig.

b) Die im vorherigen Teil berechneten Informationen reichen aus, damit A seine komplette Routing-Tabelle für das gegebene Netz erstellen kann. Der Eintrag zum Erreichen von Router C ist bereits vorgegeben. *Füllen Sie die Routing-Tabelle entsprechend Ihrer Ergebnisse aus dem vorherigen Teil aus.*

Ziel	Next Hop	Kosten
B	**C**	**6**
C	**C**	**2**
D	**C**	**5**
E	**C**	**9**
F	**C**	**3**
G	**C**	**11**
H	**C**	**12**
I	**C**	**13**

c) Die beiden prominenten Kategorien für Routing-Verfahren sind *Distance Vector* und *Link State*. Was ist der *Hauptunterschied* dieser beiden Kategorien?

Distance Vector gibt globale Informationen lokal weiter, Link State gibt lokale Informationen global weiter.

Lösung 6 (TCP) **(8+4+6+8) = 26 Punkte**

a) Da IP verbindungslos arbeitet, können keine Garantien bezüglich der korrekten Datenübertragung gegeben werden. Deshalb wird auf Transportebene oft das TCP-Protokoll eingesetzt. *Beschreiben Sie detailliert, wie TCP jeweils auf die folgenden Fehlersituationen reagiert*:

- Zwei Pakete erreichen den Empfänger in falscher Reihenfolge.
- Eine Quittung trifft erheblich verspätet beim Sender ein.

Zum ersten Fall gibt es zwei korrekte Antworten. In älteren TCP-Versionen wurden Out-of-Order-Pakete nicht zwischengespeichert. D.h., erhält TCP erst Paket 1, dann 2, dann 4, dann 3, wird 4 verworfen, 3 danach aber noch bestätigt. Nach einem Timeout auf Senderseite wird Paket 4 neu übertragen (und u.U. alles, was der Sender an folgenden Paketen noch gesendet hatte). Statt mit einem Timeout könnte man hier auch mit DUP-ACKs argumentieren.
In neueren TCP-Versionen kann der Sender Out-of-Order-Pakete zwischenspeichern. Er sortiert die Pakete in die richtige Reihenfolge und quittiert nach Erhalt von 3 direkt 4 (und damit auch 3 mit).

Eine verspätete Quittung heißt: erst trifft die Quittung für Paket 4 ein, danach die Quittung für Paket 3. Die Quittung für Paket 4 hat Paket 3 bereits mit bestätigt, insofern ist die Quittung für Paket 3 verspätet. Sie wird ignoriert.
Auch hier ist eine andere, bedingt sinnvolle Antwort unter Umständen möglich, auch wenn sie nicht sinnig ist, da ja nur _eine_ Quittung verspätet ankommt: 'verspätet' kann auch nach Ablauf des Timeouts eines Paketes sein – was allerdings nur möglich ist, wenn seitdem gar keine Quittungen mehr beim Sender eingingen, also aktuell nur eine Fenstergröße von einem Segment vorliegt. In diesem Fall wird das Paket erneut gesendet. Trifft danach die verspätete Quittung ein, wird die Wiederholung als erfolgreich angesehen. Da der Empfänger die Daten bereits bekommen hat, wird er das wiederholte Segment als Duplikat ansehen und verwerfen.

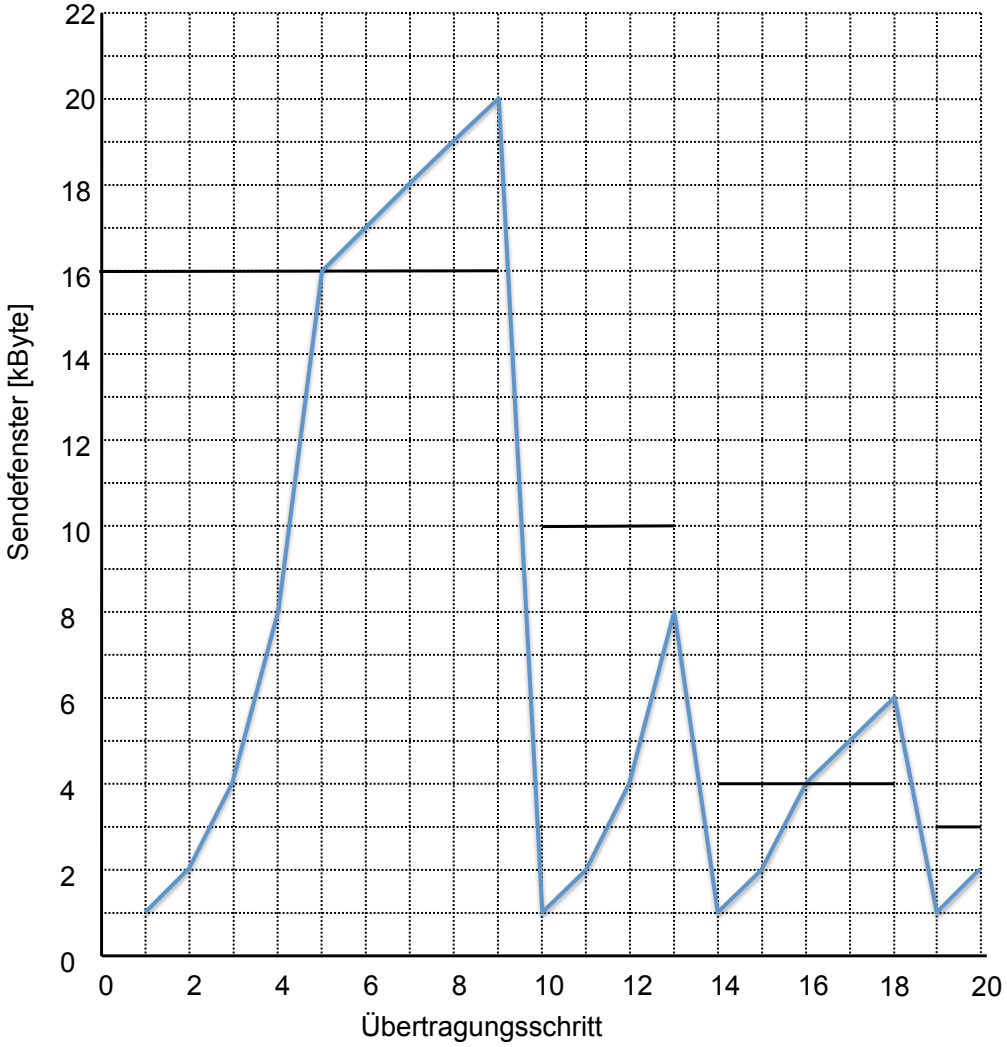
b) Angenommen, die Round-Trip-Time auf TCP-Ebene betrage 10s und die TCP-Fenstergröße der kommunizierenden Prozesse sei durch 25000 Byte limitiert. *Berechnen Sie die maximale Übertragungsrate*, die in dieser Situation erreicht werden kann.

Sliding Window sagt: sende 25.000 Byte, dann warte auf Quittungen. Erst bei Erhalt von Quittungen darf das Fenster dementsprechend weitergeschoben werden. Quittungen können erst nach der RTT kommen, also erst nach 10 Sekunden. Also kann man in 10 Sekunden 25.000 Byte schicken, somit 2.500 Byte in 1 s. Umrechnen in Bit macht: 20.000 Bit/s = 20 kBit/s.

c) Bei einem Datenaustausch zwischen zwei Kommunikationspartnern kann es neben einer Überlastung des Empfängers auch zu einer Überlastung des Netzwerks kommen. Die für diese Aufgabe betrachtete TCP-Verbindung nutzt den *Slow-Start-Algorithmus* mit einem Schwellwert (Slow Start Threshold, `ssthresh`) zur *Congestion Avoidance* von anfangs 16 kByte. Die MSS sei 1 kByte, die Window Size des Empfängers 32 kByte.

Es soll dargestellt werden, wie sich die Datenrate in diesem Szenario ändert. Dazu ist unten ein Diagramm angegeben, in welchem für die Übertragungsschritte 1 bis 20 die jeweils erreichte Übertragungsrate (ausgedrückt über die Größe des Sendefensters) dargestellt werden soll. Als ein Übertragungsschritt werde hier die Versendung der möglichen Datenmenge samt Empfang der Quittungen bezeichnet; wurden alle Quittungen des aktuellen Übertragungsschrittes erhalten, soll im nächsten Übertragungsschritt wieder die nun mögliche Datenmenge versendet werden. Beim 9., 13. und 18. Übertragungsschritt finde ein Timeout statt, der vom Sender als Netzüberlastung interpretiert wird.

Zeichnen Sie für die Übertragungsschritte 1 bis 20 jeweils die Größe des Sendefensters sowie den Threshold in das Diagramm ein.



d) TCP arbeitet bidirektional. Welche TCP-Headerfelder werden für die *Rolle als Empfänger* in der Datenaustauschphase verwendet? *Geben Sie vier Felder an und begründen Sie, warum/wofür der Empfänger sie verwendet.*

Jede Menge, z.B.:

- Source-Port in empfangenen Daten, da man dorthin seine Quittungen schicken muss. Bzw. beim Senden von Quittungen füllt man hier seine eigene Adresse ein.
- Destination-Port muß beim Empfangen von Daten geprüft werden, um die Daten richtig zuordnen zu können.
- Sequenznummer muß geprüft werden, um festzustellen, ob dies das nächste erwartete Segment ist und eine Quittung geschickt werden kann.
- Entsprechend wird die Quittungsnummer beim Versenden von Quittungen benötigt.
- ... und damit zusammen das ACK-Flag, damit die andere Seite die Quittungsnummer betrachtet.
- Das Feld HL muß beim Empfangen geprüft werden: gibt es Optionen? Wo fängt der Datenteil an?
- Das WIN-Feld wird in Quittungen benötigt: wir teilen dem Sender mit jeder Quittung mit, wie viel Buffer noch zur Verfügung steht.
- Die Checksum wird benötigt – bevor wir Daten verarbeiten und eine Quittung senden, testen wir das Paket auf Korrektheit.

Nicht korrekt sind SYN- und FIN-Flag - diese betreffen Verbindungsauf- und -abbau, Sender- und Empfängerrolle gibt es nur beim Datenaustausch.

Lösung 7 (Sicherheit) **(6+3+5) = 14 Punkte**

a) *Berechnen Sie einen geheimen Schlüssel unter Verwendung des Diffie-Hellman-Algorithmus'*. Nutzen Sie $p = 11$ und $g = 3$. Verwenden Sie als Geheimzahlen der Kommunikationspartner kleine Zahlenwerte.

1.) Alice wählt $a = 3$ und berechnet $A = g^a \bmod p = 3^3 \bmod 11 = 5$.

2.) Alice sendet g, p und A an Bob (oder nur A, falls der Rest vorher schon bekannt war).

3.) Bob wählt $b = 2$, berechnet $B = g^b \bmod p = 3^2 \bmod 11 = 9$ und noch dazu den Schlüssel $K = A^b \bmod p = 5^2 \bmod 11 = \mathbf{3}$. Außerdem sendet Bob seinen Wert B an Alice.

4.) Nun kann Alice auch den Schlüssel berechnen: $K = B^a \bmod p = 9^3 \bmod 11 = \mathbf{3}$.

b) Sie verwenden *RSA* als asymmetrisches Verschlüsselungsverfahren und haben $p = 7$ and $q = 3$ gegeben. *Ist* $< 3, 21 >$ *ein gültiger geheimer Schlüssel?* Begründen Sie Ihre Antwort.

Berechne zunächst $\phi(n)$ mittels $n = p * q = 21$: $\phi(n) = (p-1) * (q-1) = 12$.
Wähle e relativ prim zu 12 sowie als multiplikativ Inverses zu $d = 3\ mod\ 12$. D.h.: existiert ein e mit $e * 3 mod 12 = 1$? Da 3 Faktor von 12 ist, gibt es keins. Man kann die Lösung auch schon abkürzen, indem man direkt sagt, dass die 3 bereits teilerfremd zu 12 sein müsste.
Wer das nicht weiß, muss alle Werte auf der Suche nach einem Inversen durchtesten (was nicht lange dauert, da es nicht viele sind) und findest nix. Oder man führt overkillmäßig den erweiterten euklidischen Algorithmus durch, aber der ist hier völlig überflüssig.

c) Die Firma SecureSys bietet ihre Sicherheitssoftware S auf ihrer Webseite zum Download an. Der Webserver hat ein Zertifikat mit einem öffentlichen Schlüssel; der zugehörige private Schlüssel ist sicher hinterlegt. Der Systemadministrator schlägt folgende Authentifizierungsmethode für den vertrauenswürdigen Download der Software vor: Zusammen mit Software S wird der folgende Wert im Download zur Verfügung gestellt: $H(S||KU_W)$.

Es gilt die folgende Syntax:

- S: Software
- KR_W: privater Schlüssel des SecureSys-Webservers
- KU_W: öffentlicher Schlüssel des SecureSys-Webservers
- $H(M)$: kryptographischer Hash von M
- $||$: Konkatenation zweier Bitfolgen

Ist die vorgeschlagene Methode geeignet, um den Download zu authentifizieren? Wenn ja, warum? Und wenn nein, was sollte stattdessen getan werden?

Die Methode ist nicht geeignet, denn jeder der die Hashfunktion und den öffentlichen Schlüssel kennt (und das sind alle), kann die Authentifizierungsinformationen fälschen. Der Hashwert sollte mit dem privaten Schlüssel verschlüsselt werden! Wobei da schon der Hashwert über S reicht.