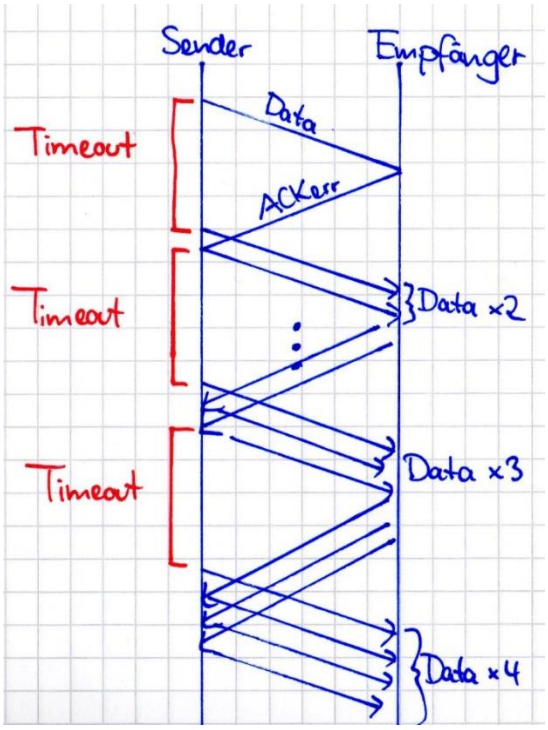
Lösung 10: Zuverlässige Datenübertragung

Aufgabe 2: Zuverlässige Datenübertragung

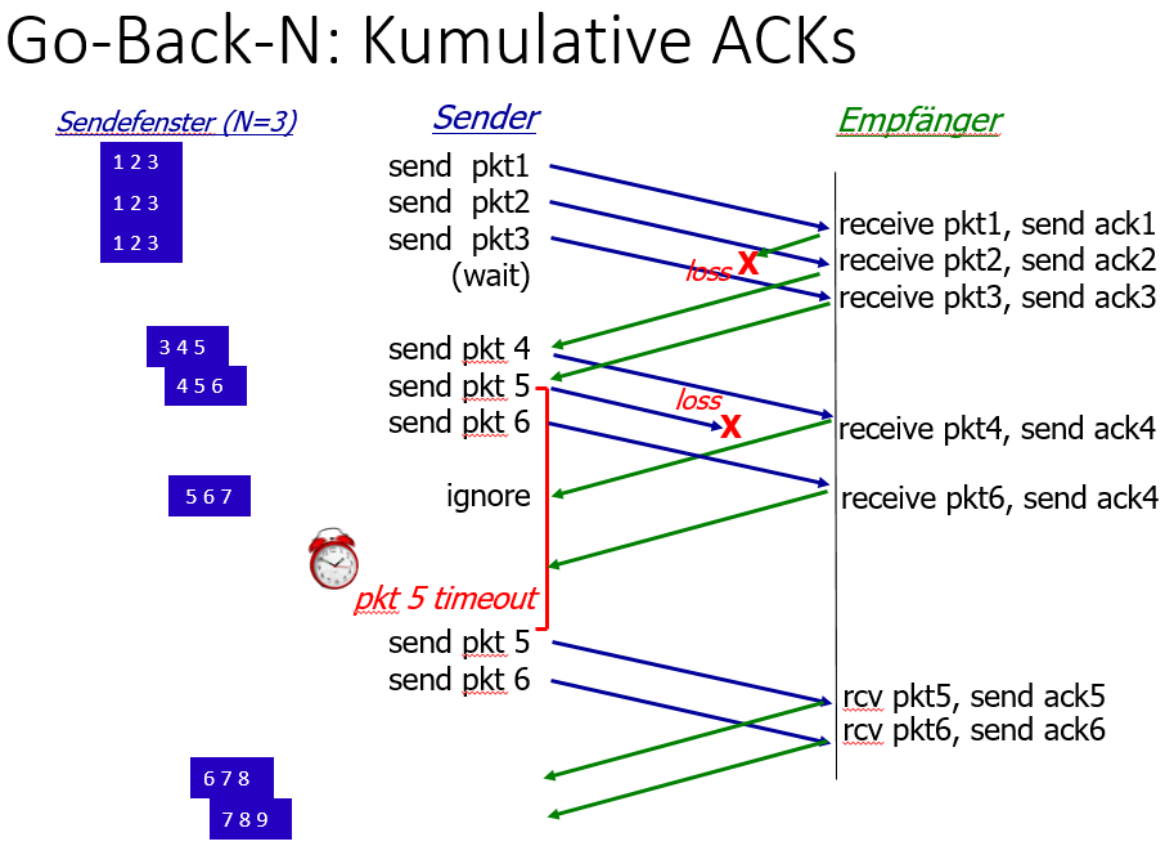
a) In Abbildung 1 nimmt die Anzahl der gleichzeitigen Pakete linear zu, was sicherlich ungünstig ist. Das liegt darin, dass der Sender beim Verfahren (i**) immer sofort** eine Retransmission startet, selbst dann wenn der Sender gerade eben zuvor erst eine Retransmission auf den Weg geschickt hat. Der Sender interpretiert ein korruptes ACK immer als einen eindeutigen Hinweis, dass eine Retransmission nötig ist. Man überlegt sich leicht, dass im Fall (ii) deutlich weniger Nachrichten gleichzeitig unterwegs wären, weswegen diese Variante im konkreten Fall vorzuziehen ist.



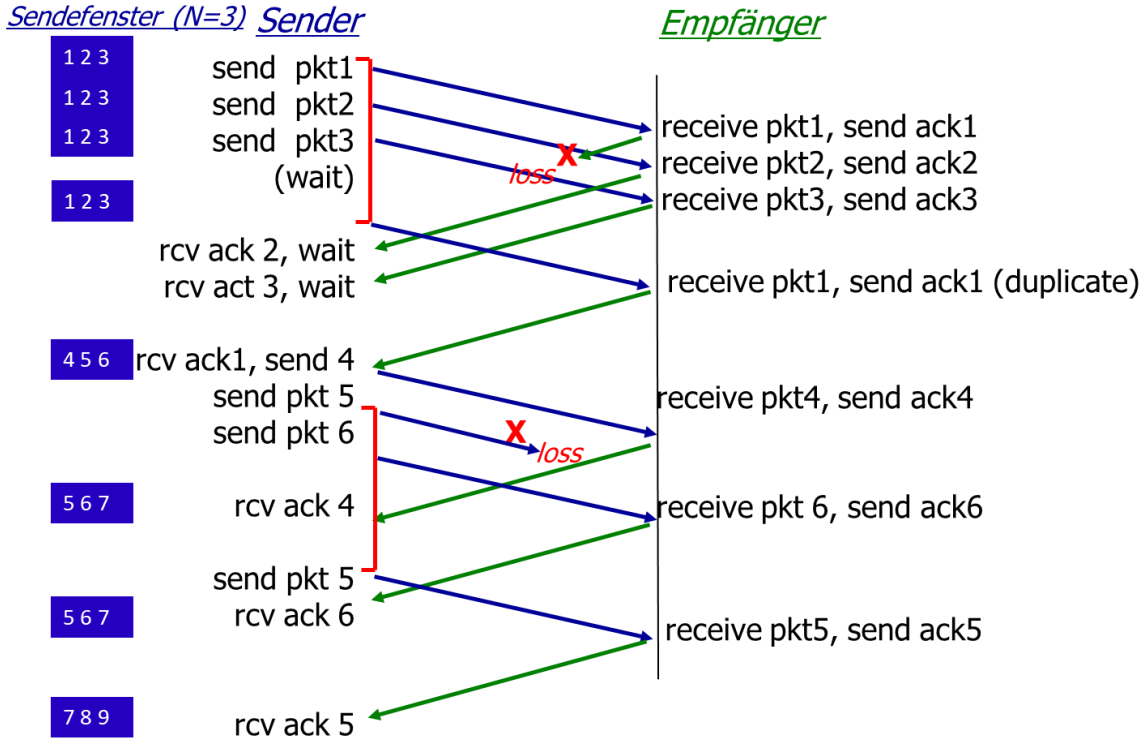
Aufgabe 2: Go-Back-N, Selective Repeat

Hinweis: Im fiktiven Protokoll Version 3.0 der Vorlesung bestätigt ein ACK x immer, dass das bereits erhaltene Paket mit der höchsten Sequenznummer die Sequenznummer x trägt. Später bei TCP ist es leicht anders! Hier werden nicht Pakete sondern Bytes aus dem Bytestrom bestätigt. Ein ACK x bestätigt bei TCP, dass alle Bytes mit der „Nummer“ < x bereits erhalten wurde, das nächste erwartete Byte ist x.

a) Siehe Diagramm. Die Annahme hier ist, dass das ACK=2 beim Sender eintrifft, bevor der Timeout für Paket 1 ausläuft. Andere Annahmen sind erlaubt, solange die Gesamtlösung schlüssig ist.



b) Siehe Diagramm.



c) Die Tatsache, dass der Empfänger als nächstes das Paket mit der Sequenznummer 500 erwartet bedeutet, dass er bereits das Paket 499 erhalten und bestätigt hat, sowie alle vorangegangenen Pakete. Nun ist nur nicht klar, ob der Sender die ACKs vom Empfänger für Paket 499, 498, 497, und 496 auch bereits erhalten hat. Das ACK für 495 muss er erhalten haben, ansonsten hätte der

Sender das Datenpaket 499 wegen der Größe 4 des Sendefensters gar nicht an den Empfänger senden dürfen. Man unterscheidet am besten 2 Fälle:

- Alle ACKs 499, 498, 497, 496 sind auch beim Empfänger eingetroffen: Dann ist das Sendefenster [500; 503].
- Keine der ACKs ist beim Empfänger eingetroffen: Dann ist das Sendefenster [496; 499]

Dies sind die beiden Extremfälle. Irgendwo „dazwischen“ kann das Sendefenster liegen.

d) Es gilt: $d_{trans} = \frac{L}{R} = 12\ us$

$d_{trans} + 2 * 15\ ms$ nach Beginn des Sendevorgangs, erhält der Sender die Bestätigung für sein Datenpaket. In diesem Zeitintervall ist der Sender nur für den Zeitraum von d_{trans} beschäftigt. Ohne Pipelining beträgt seine Auslastung also:

$$Auslastung = \frac{d_{trans}}{d_{trans} + 2 * 15\ ms}$$

Durch Einsatz von Pipelining (Sendefenster N, N Nachrichten können auf Reise sein) ist er besser beschäftigt:

$$0{,}98 = Auslastung = \frac{N \cdot d_{trans}}{d_{trans} + 2 * 15\ ms}$$

Auflösen nach N ergibt:

$$N = 0{,}98 \cdot \left(\frac{d_{trans} + 2 * 15\ ms}{d_{trans}}\right) = 2451$$

Die Fenstergröße müsste als theoretisch 2451 betragen.

Aufgabe 3: Scanning mit `nmap`

c) Kommando: `nmap -sP 192.168.1.0/24.` Im Wireshark erkennt man, dass anstelle von ICMP Pings oder anderer Methoden im konkreten Fall **ARP Requests** zur Erkennung von aktiven Hosts eingesetzt werden. Antwortet ein Host auf einen ARP Broadcast, so ist das ein deutliches Indiz, dass dieser Host aktiv ist. Die Methode funktioniert nur im lokalen LAN, ist aber dann deutlich effizienter. Ggfs. würde NMAP bei entfernten Subnetzen andere Methoden automatisch einsetzen.

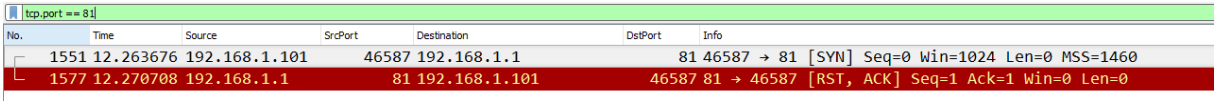
```
Starting Nmap 7.70 ( https://nmap.org ) at 2018-06-07 10:00 Mitteleuropäische Sommerzeit
Nmap scan report for 192.168.1.1
Host is up (0.0031s latency).
MAC Address: C0:56:27:4F:76:07 (Belkin International)
Nmap scan report for 192.168.1.101
Host is up.
Nmap done: 256 IP addresses (2 hosts up) scanned in 5.95 seconds
```

Details, siehe Wireshark Trace host-scan.png

d) Kommando: `nmap -sS 192.168.1.1`

Details, siehe port-scan.pcapng (Filter: tcp.flags.syn and ip.addr ==192.168.178.34)

Im konkreten Fall sendet Wireshark schrittweise sogenannte TCP SYN Pakete an den Zielhost und zwar ein TCP SYN Paket für jeden Zielport. Falls der Port geschlossen ist, antwortet der Zielhost mit einem sogenannten TCP Reset Paket. Wichtig: Ein TCP Reset Paket muss beim Scanner nicht ankommen, z.B. wegen vorhandener Firewalls. Ein fehlendes TCP Reset ist also nicht zwingend ein Zeichen für einen geschlossenen Port, aber zumindest ein Indiz.



Falls der Port offen ist, wird ein TCP SYN ACK Paket vom Zielhost an den Scanner zurückgesendet.