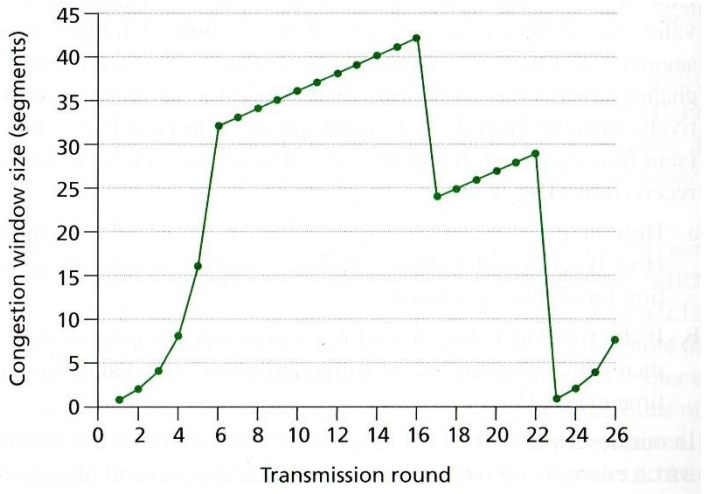
Übung 12: Congestion Control, NAT, SSH Tunneling

Aufgabe 1: TCP Congestion Control

a) Was ist der Unterschied zwischen *Flow Control* und *Congestion Control*?



TCP Größe des Congestion Windows (`cwnd`) als Funktion der Zeit

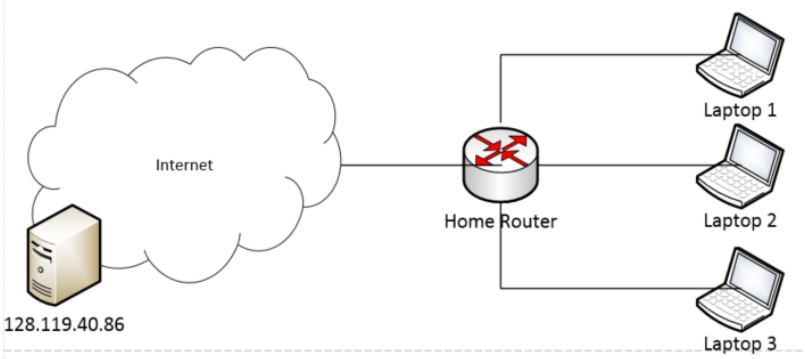
Quelle: Kurose&Ross

Die Abbildung zeigt die Größe des Congestion Windows (`cwnd`) nach jeder „Übertragungsrunde“(RTT Time). Es wird TCP Reno und Fast Retransmit eingesetzt. Ferner werden immer maximal große Pakete versendet (*Maximum Segment Size*).

b) In welchen Zeitintervallen ist TCP Slow Start aktiv?
c) In welchen Zeitintervallen wird TCP Congestion Avoidance verwendet?
d) 16. Runde: Wird Paketverlust durch Timeout oder durch 3 ACK-Duplikate erkannt?
e) 23. Runde: Wird Paketverlust durch Timeout oder durch 3 ACK-Duplikate erkannt?
f) Welchen Wert hat `ssthres` in der 1. Runde?
g) Welchen Wert hat `ssthres` in der 18. Runde?
h) Welchen Wert hat `ssthres` in der 24. Runde?
i) In welcher Runde wird das 17. Paket gesendet?

Aufgabe 2: Network Address Translation (NAT)

3 Laptops in einem Heim-Netzwerk sind über einen Home Router und NAT mit dem Internet verbunden. Die öffentliche IP Adresse des Routers sei 24.34.112.235, innerhalb des Heim-Netzwerks dürfen ***ausschließlich*** Adressen aus dem privaten IP Adressbereich 192.168.0.0/24 gewählt werden.



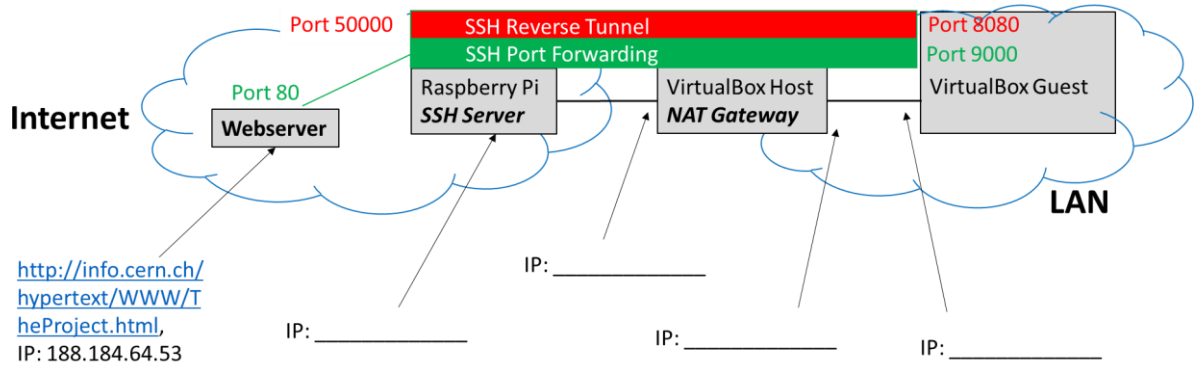
a) Weisen Sie *allen* Interfaces innerhalb des Heim-Netzwerks manuell eine gültige IP Adresse zu. Tragen Sie auch die öffentliche IP Adresse des Routers in die Zeichnung ein.

b) Nehmen Sie an, dass zu einem bestimmten Zeitpunkt alle Laptops im Heim-Netzwerk gleichzeitig jeweils 2 HTTP Verbindungen zum Web Server 128.119.40.86 unterhalten. Wie könnten eine mögliche NAT-Tabelle des Home Routers dann für diesen Zeitpunkt aussehen?

WAN Seite/Internet		***LAN Seite / Heim-Netzwerk***	
IP Adresse	Port	IP Adresse	Port

Aufgabe 3: SSH Tunneling

a) Gegeben sei die folgende Topologie. Sie kontrollieren nur den VirtualBox Host und den VirtualBox Gast[1]. Tragen Sie in die Zeichnung die 4 fehlenden IP Adressen ein. Die IP des Raspberry Pi wird in der Übung bekanntgegeben.



b) Tool `curl`: Mit `curl` kann man per Kommandozeile Webseiten aufrufen. Testen Sie, dass die erste Webseite des Internets vom Gast aus abrufbar ist:

```
curl --header 'Host: info.cern.ch' 'http://188.184.64.53:80/
                                        hypertext/WWW/TheProject.html'
```

c) Regulärer SSH Zugriff: Bauen Sie vom Gast zum Raspberry Pi eine verschlüsselte SSH Verbindung („***SSH Tunnel***“) auf. Sie können nun von außen auf dem Raspberry Pi arbeiten als säßen Sie selbst vor dem Raspberry. Pingen Sie die IP des Gastes. Ist das erfolgreich? Warum?

- `ssh pi@<ip>` (Passwort: `raspberry`)
- `ping <ip>`

d) SSH Port Forwarding: Nehmen Sie an, das NAT Gateway blockiere alle ***ausgehenden*** Anfragen an Port 80. Über den SSH Tunnel kann der Gast jedoch die Daten von einem Dritten (== SSH Server) anfordern lassen, der die Antwort dann über einen SSH Tunnel an den Gast zurückleitet. So umgehen Sie die Blockade. Stellen Sie dieses Szenario nach:

- Emulation der Blockade: Auf Gast (!!) alle ausgehenden Pakete zu Port 80 blockieren mit dem folgenden Kommando: `sudo iptables -A OUTPUT -p tcp --dport 80 -j REJECT`
- Testen, dass der Aufruf der Webseite mit `curl` (siehe oben) scheitert.
- SSH Tunnel aufbauen, so dass Daten vom lokalen Port 9000 über SSH Tunnel zur IP 188.184.64.53 Port 80 weitergeleitet werden. Die Antwort wird dann korrekt zurückgeleitet. Sie benötigen die **Option `-L`**[2]: `ssh pi@<ip> -L <?????>`
- Rufen Sie die Webseite ab. *Wichtig:* IP/Port des `curl`-Kommandos (siehe oben) anpassen!

[1] Es wird die VM des Moduls Betriebssysteme empfohlen.
[2] Internet-Recherche: https://linux.die.net/man/1/ssh

- „Blockade entfernen“: `sudo iptables -F`

e) SSH Reverse Tunnel: Sie sitzen hinter einem NAT Gateway und möchten einen Server betreiben, der aus dem Internet erreichbar ist. Sie können dem SSH Server mitteilen, dass der SSH Server **ankommende** TCP Verbindungen für einen bestimmten Port immer an einen bestimmten Port des Gastes weiterleitet. Simulieren Sie dieses Szenario:
   - Webserver auf Port 8080 des Gastes betreiben: `python -m SimpleHTTPServer 8080`
   - Testen Sie, dass die Webseite lokal erreichbar ist (aber nicht von aussen): http://127.0.0.1/8080
   - SSH Reverse Tunnel aufbauen: Anfragen an Port 50000 des SSH Servers sollen an Port 8080 des Gastes weitergeleitet werden. **Option `-R`**, siehe [2]
   - Testen Sie mit dem Webbrowser Ihres Handys, dass die Webseite von außen erreichbar ist.