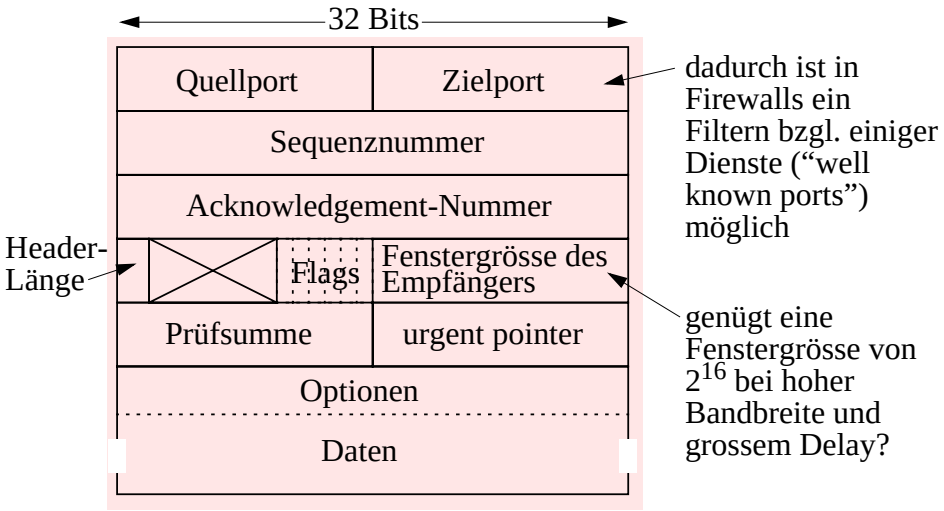
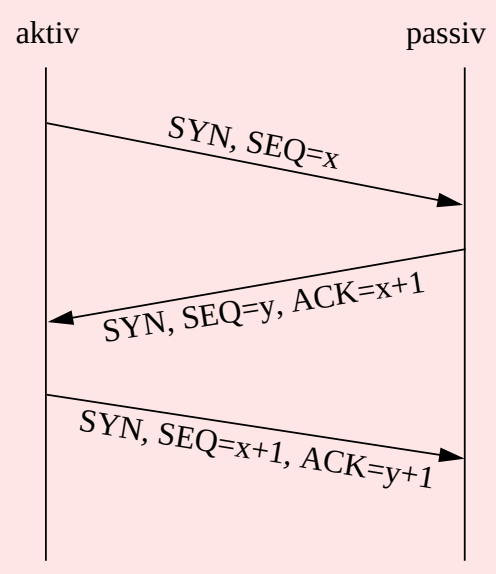
Der TCP-Header



- Vorher (im IP-Header) stehen die IP-Adressen
- Jedes Byte wird numeriert (für sliding window)
  - Sequenznummer bzw. Ack.-Nummer
  - wrap-around der Sequenznummern bei 100 Mb/s nach 6 min möglich!
- Ack.-Nr. und Fenstergrösse dienen der Rückmeldung vom Empfänger zum Sender (ggf. huckepack mit Nutzdaten in Rückrichtung versandt)
- Mit den Flags kann folgendes ausgedrückt werden:
  1) URG: “urgent”: dringende Daten (z.B. cntrl-C)
  2) ACK: Acknowledgement-Nummer ist gültig
  3) PSH: Empfänger soll nicht puffern (z.B. <CR> bei Zeilenende)
  4) RST: Reset der Verbindung (nach einem erkannten Problem)
  5) SYN: Aufbau einer Verbindung (Synchronisieren erwarteter Sequenznummern)
  6) FIN: Beenden einer Verbindung

Verbindungsauf- und abbau

- Aufbau der Duplex-Verbindung durch 3-fach-Handshake



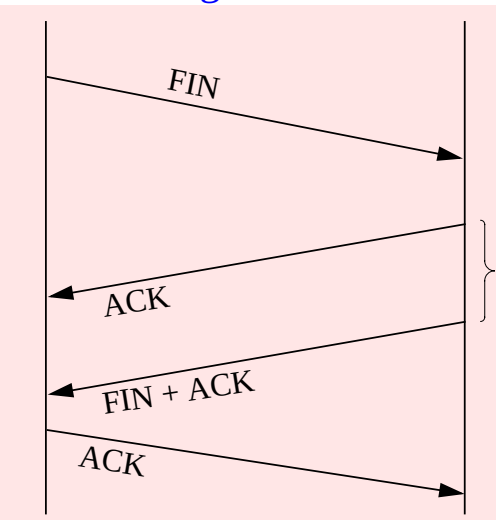
Austausch und Bestätigung initialer Sequenznummern, um alte von neuen Daten auch nach einem Rechnerabsturz zu unterscheiden

Hier aktiver Partner (“Client”) und passiver Partner (“Server”)

Gleichzeitiger Verbindungsaufbau von zwei aktiven ist auch möglich (resultiert aber in einer einzigen Verbindung)

Denkübung: Was kann geschehen, wenn Kontrollnachrichten verloren gehen oder nach langer Zeit “aus dem Nichts” auftauchen?

- Verbindungsabbau durch vier Kontrollnachrichten

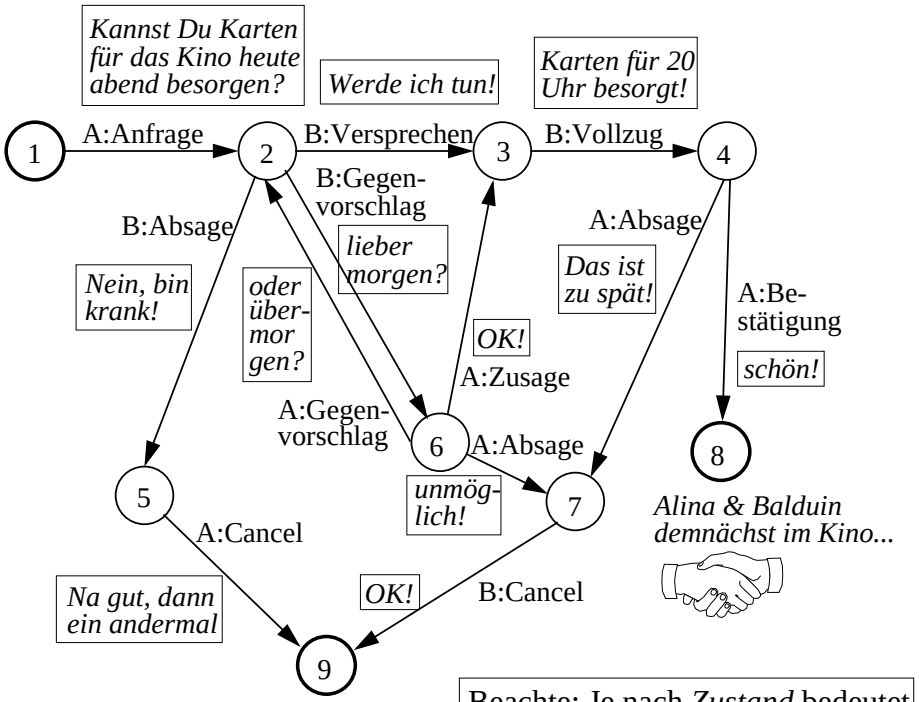


Denkübung: Wie kann sich ein Kommunikationspartner sicher sein, dass der andere die Verbindung ebenfalls abbaut, wenn Kontrollnachrichten (z.B. ACK) verloren gehen können?

Hier können noch Daten gesendet gesendet werden

Es gibt auch den Verbindungsabbruch (statt Abbau); dann können ggf. gesendete Daten verloren gehen

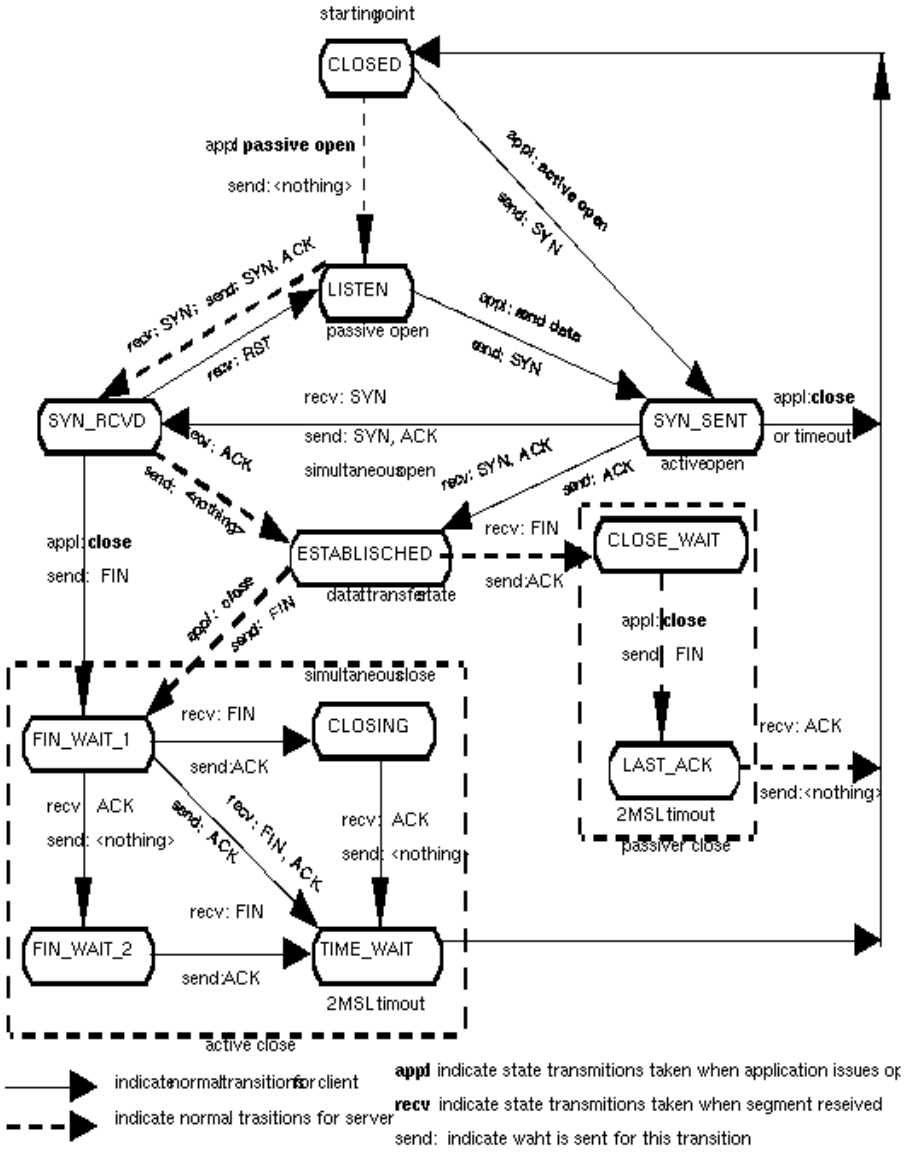
Zustandsübergangsdiagramme für Protokolle



- "Konversationszustände" --> *endl. determ. Automat*
- Ggf. *Missverständnisse* bei Nachrichtenverlust!
- Koordination von Handlungen --> *Protokolle*
- vgl. *Sprechakttheorie*: Allgemeines Schema für viele möglichen "Sprechaktfolgen" (z.B. 1,2,6,2,3,4,8)

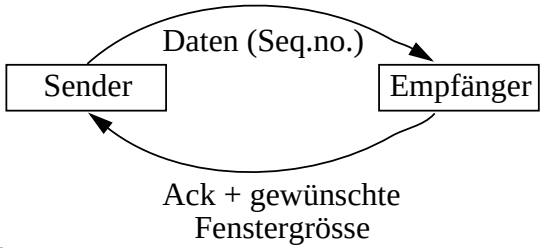
TCP-Zustandsdiagramm

Vgl. dazu die Literatur (z.B. Tanenbaum p. 532; Fig. 6-28)



TCP: Fluss- und Laststeuerung

- Sliding window-Protokoll



- Fenstergrösse adaptiv
  - es gibt zwei Fenstergrössen:
    1) Wunsch des Empfängers
    2) Grösse des "congestion window", das durch TCP selbst aufgrund der Beobachtung der Netzlast dynamisch verändert wird
  - Sender richtet sich nach dem Minimum der beiden Werte
  - Maximum ist i.a. auch durch die Puffergrösse beim Einrichten eines sockets bestimmt
  - in TCP-Paketen nur 16 Bits für die Fenstergrösse vorgesehen --> Skalierungsfaktor vor Start der Kommunikation "aushandeln"
  - ideal wäre das Bandbreite-Delay-Produkt (wieso?)

z.B. mittels ping ermittelbar

Denkübung: einfache Zeit oder round-trip-Zeit?

- Timeouts für Retransmissionen adaptiv
  - richtigen Wert zu finden ist eine Kunst!
- Bestätigungen durch Acks sind kummulativ
  - kummulative Acks bei Hochgeschwindigkeit allerdings problematisch, selektives Ack / retransmit wäre effizienter

Congestion Window bei TCP

- Problem der Aufschaukelung von Netzüberlastungen:
  Netzüberlastung --> Timeouts --> Paketwiederholungen --> noch mehr Last
- TCP ist an sich "selbsttaktend" durch die Verwendung von Acknowledgements bei sliding window
- Um einer Netzüberlastung gegenzusteuern, wird bei Überlast das congestion window verkleinert
- Indikatoren für eine Netzüberlastung:
  - timeout bzgl. eines erwarteten Ack
  - Empfang einer "source quench"-ICMP-Nachricht

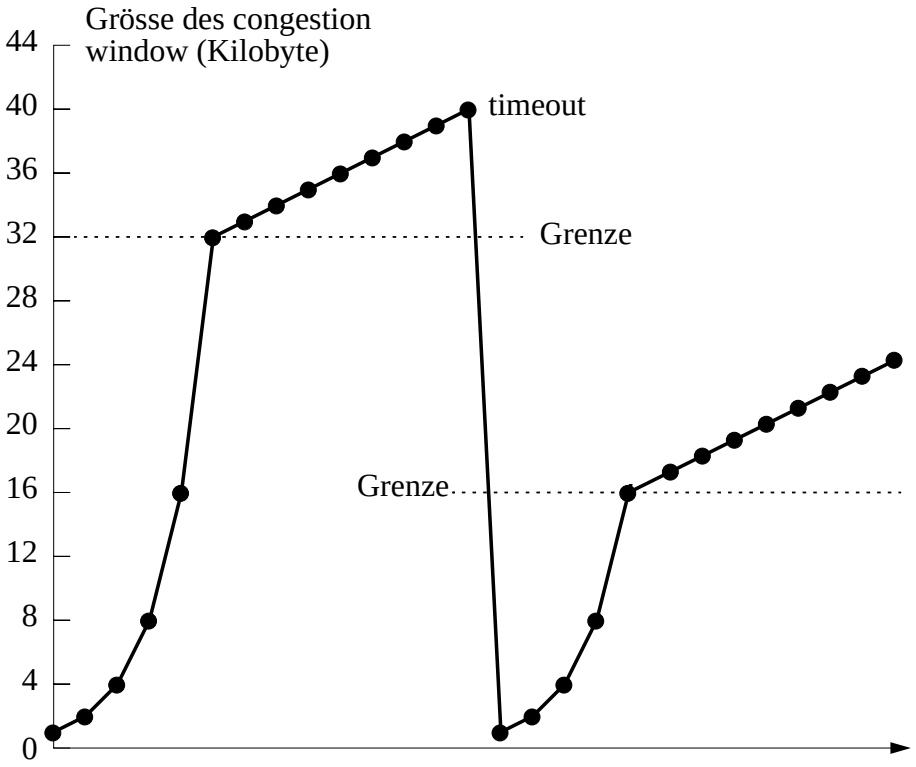
- Es gibt verschiedene Strategien zur dynamischen Veränderung des congestion window, u.a.:

1) "additive increase / multipicative decrease"
   - bei jedem timeout die Fenstergrösse halbieren
   - bei Erfolg (Ack bzgl. eines TCP-Segments kommt vor Ablauf des timeouts an), Fenstergrösse um ein festes Inkrement erhöhen

2) "slow start"
   - bei Erfolg Fenstergrösse verdoppeln
   - jedoch nur bis zu einem Grenzwert, ab dann linear vergrössern
   - nach einem timeout den Grenzwert auf die Hälfte der ggw. Fenstergrösse setzen und aktuelle Fenstergrösse auf einen kleinen Wert setzen

slow

Slow start



Timeout-Bestimmung bei TCP

- Problem: Varianz der Übertragungszeiten ist sehr gross

- Man arbeitet mit einem Schätzwert RTT für die round-trip-Zeit, der laufend (gleitend) angeglichen wird:

$$RTT = \alpha\, RTT + (1-\alpha)\, M$$

wobei M die gemessene round-trip-Zeit bzgl. des letzten Acknowledgements ist und α typw. auf 7/8 gesetzt wird

- Der timeout-Wert wurde dann ursprünglich so bestimmt:

$$timeout = 2\, RTT$$

- Spätere TCP-Implementierungen verwenden stattdessen:

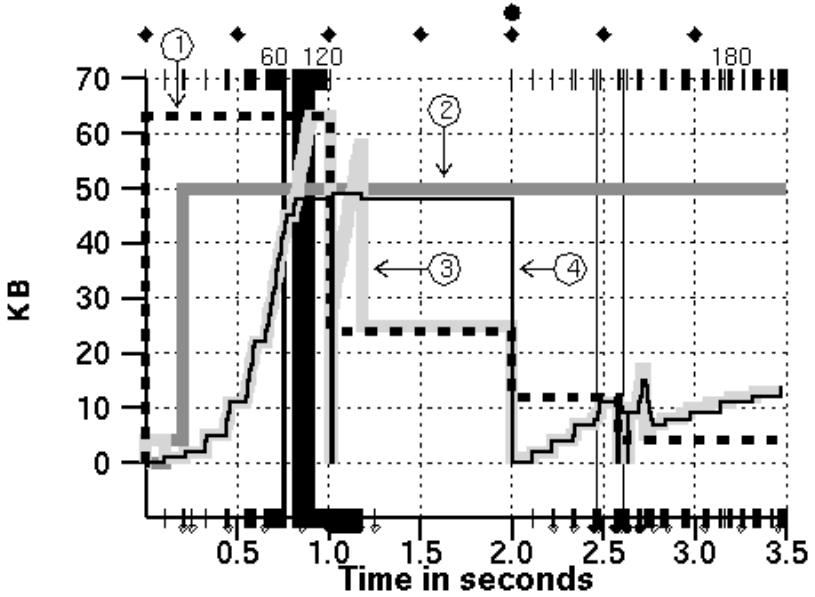
$$timeout = RTT + V$$

wobei V ein Wert ist, der aus der beobachteten Varianz der letzten round-trip-Messungen hervorgeht (Motivation: bei grösserer Varianz sollte der timeout grösser sein)

- Denkübung: Welche negativen Konsequenzen hat eigentlich ein zu grosser / zu kleiner timeout-Wert?

TCP-Benchmarks

1) Markierung auf der x-Achse: Ack wird empfangen
2) Markierungsstrich am oberen Bildrand: IP-Paket wird versendet
3) Zahl *n* am oberen Bildrand: Wann das *n*-te Kilobyte versendet wird
4) Rauten am oberen Bildrand: Nur alle 0.5 Sekunden wird überprüft, ob ein timeout abgelaufen ist
5) Punkt am oberen Bildrand: Retransmission wegen abgelaufenem timeout

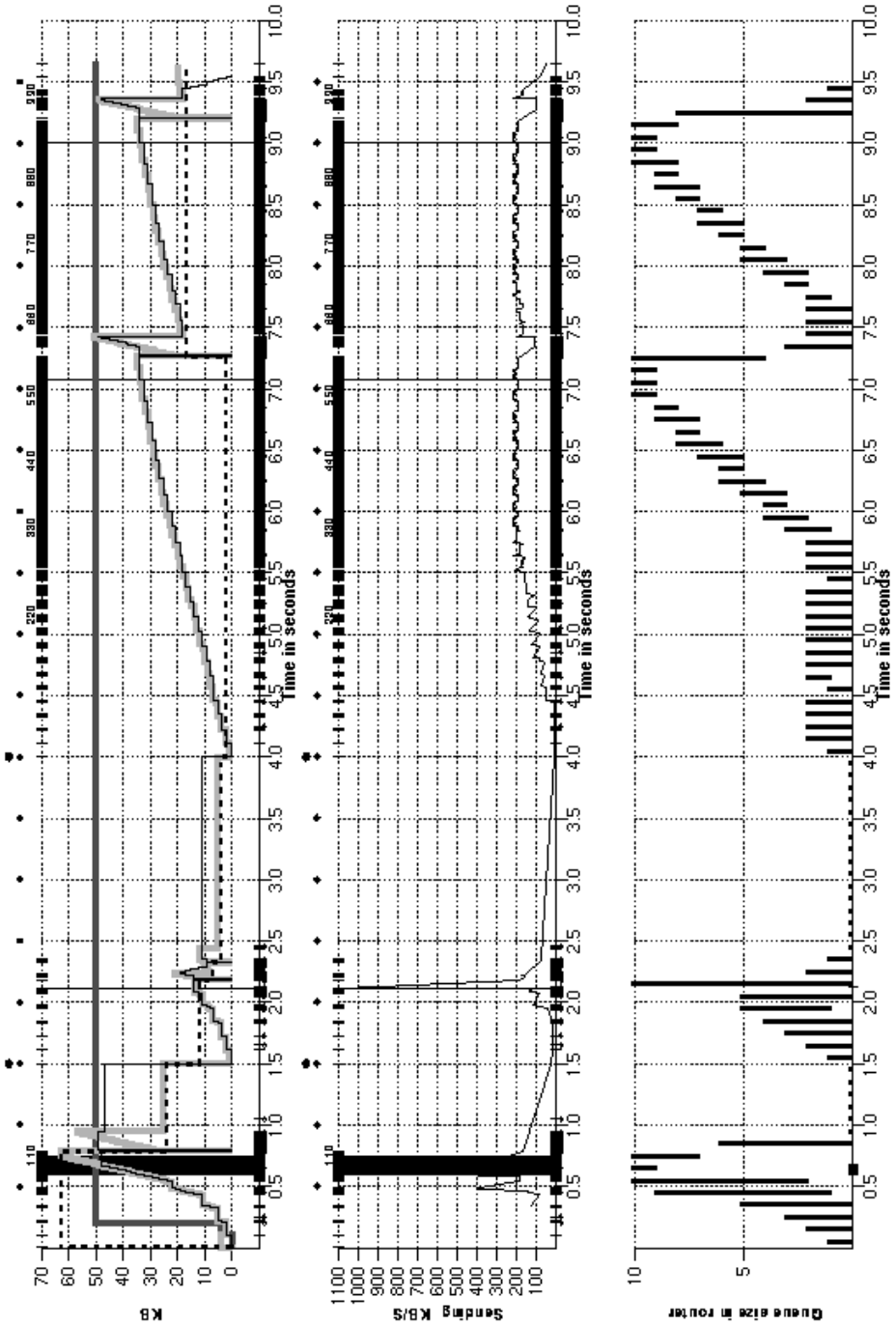


(1) gestrichelte Linie: Grenzwert exponentielle / lineare Fenstervergrösserung
(2) dunkelgraue Linie: Grösse des Sendefensters
(3) hellgraue Linie: Grösse des congestion windows
(4) dünne Linie: Anzahl der gesendeten, aber noch nicht bestätigten Bytes

---

Nächste Folie (oberes Bild) zeigt dies über einen längeren Zeitraum; im mittleren Bild ist der effektive Durchsatz angegeben; im unteren Bild die Länge einer Router-Warteschlange (max. 10). Zwischen 5.5 und 7 wird das congestion window vergrössert, der Durchsatz (mittleres Bild) bleibt jedoch gleich (offenbar ist die max. Netzbandbreite erreicht!). Die höhere "Sendeleistung" muss von den Puffern des Routers abgefedert werden!

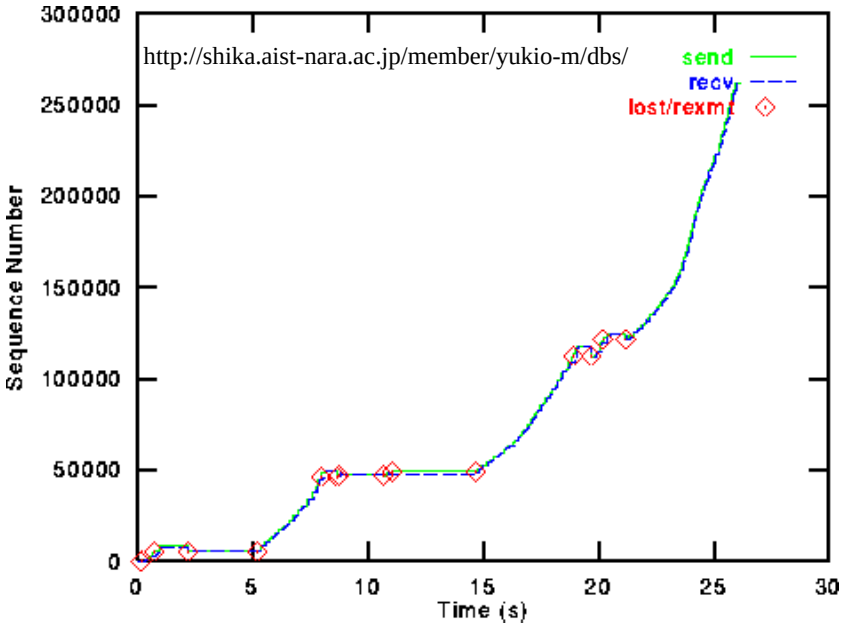
Quelle: http://excalibur.usc.edu/research/vegas/doc/vegas.html



TCP-Benchmark (2)

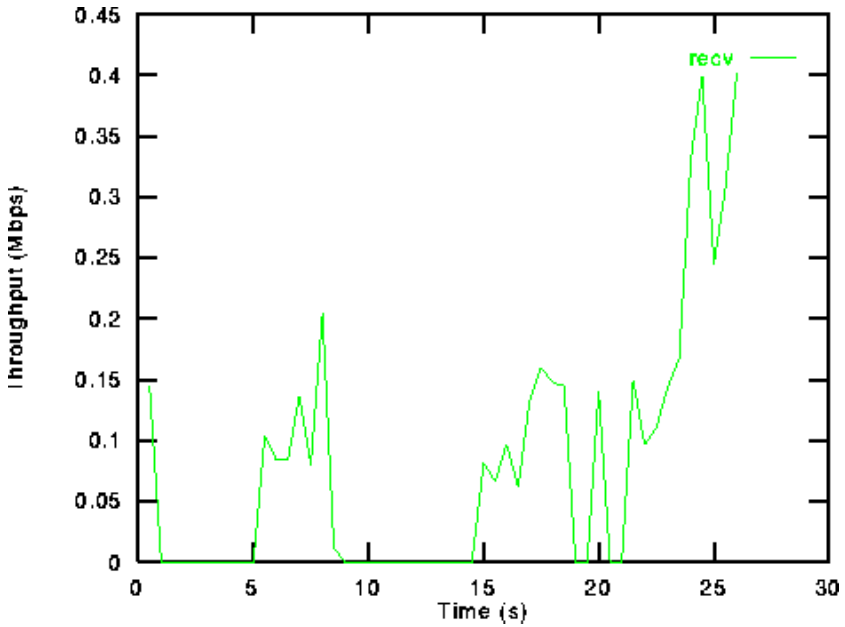
- Terrestrische Verbindung mit 1.5 Mb/s Bandbreite
  - Weitverkehrsverbindung über mehrere 100 km (typ. für Internet)
  - Beobachtung: TCP wiederholt ca. 1% aller TCP-Segmente
  - max. Fenstergrösse 8196 Byte
  - Pakete von 1024 Byte
  - typisches Szenario: Dateitransfer

- *Sequenznummer* der Bytes über die Zeit gemessen:

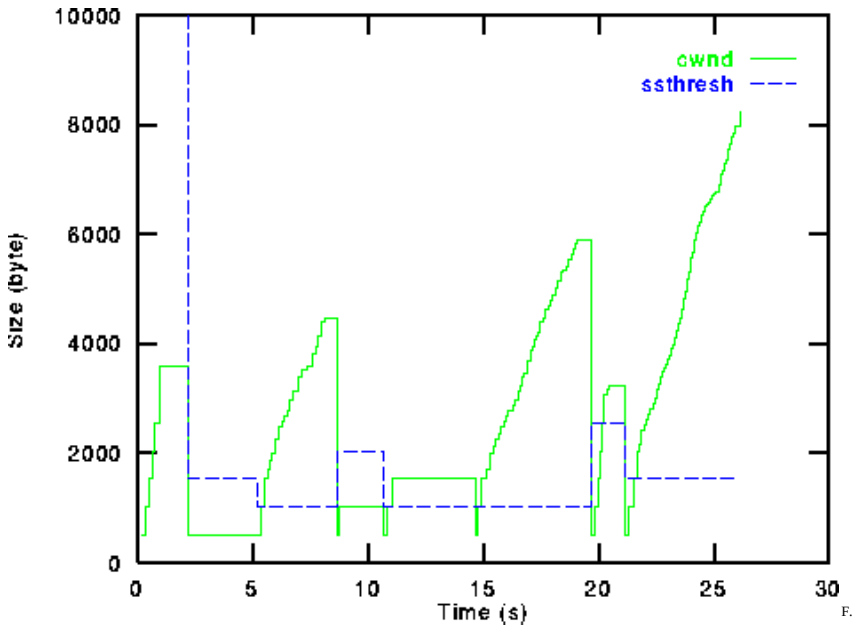


  - Anstieg ist nicht linear --> keine gleichmässige “Geschwindigkeit”
  - lange Phasen, wo nichts geschieht (“stalled”)
  - sind Retransmissionen Folge oder Ursache für die Probleme?

- *Durchsatz* im Mittel nur ca. 80 kb/s --> 5% Effizienz:

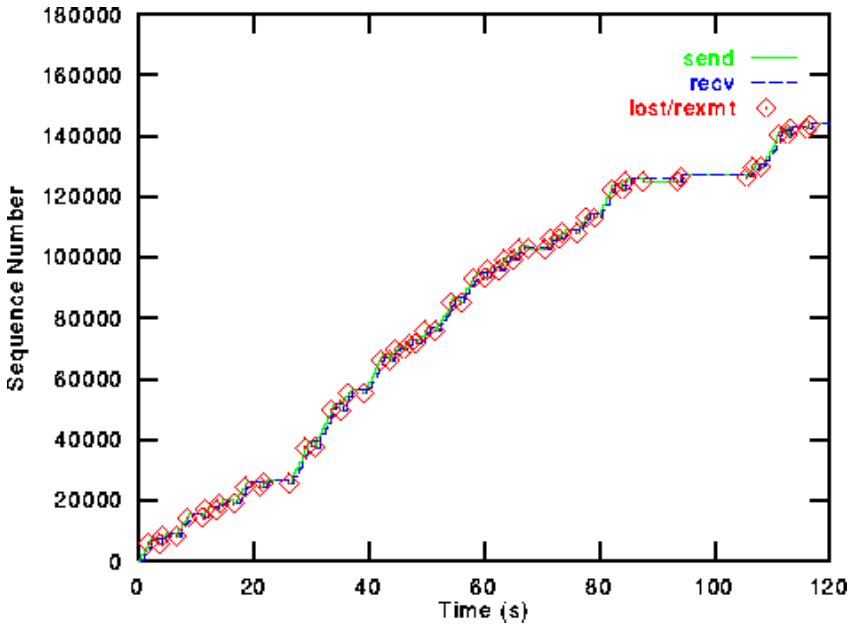


- *Congestion Window* fällt oft auf einen kleinen Wert:

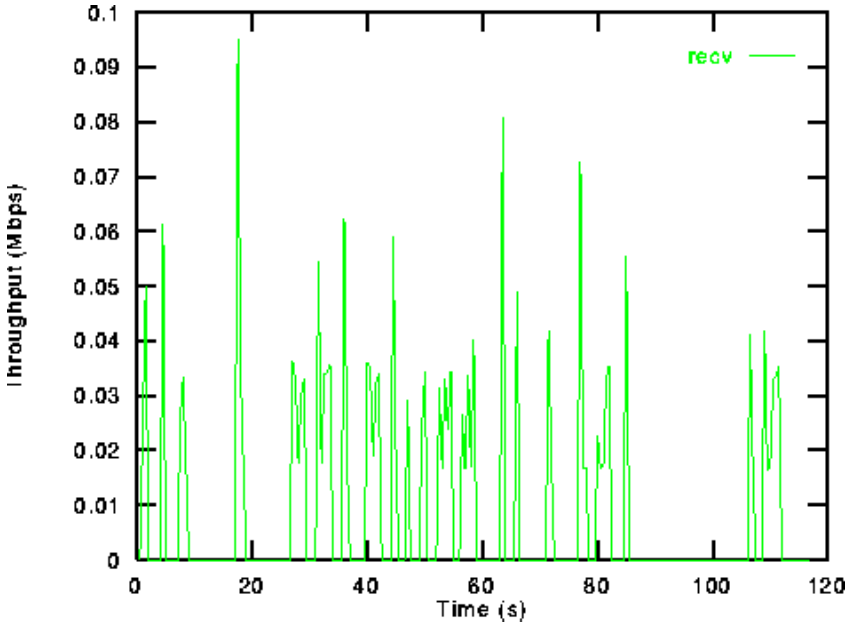


TCP-Benchmark (3)

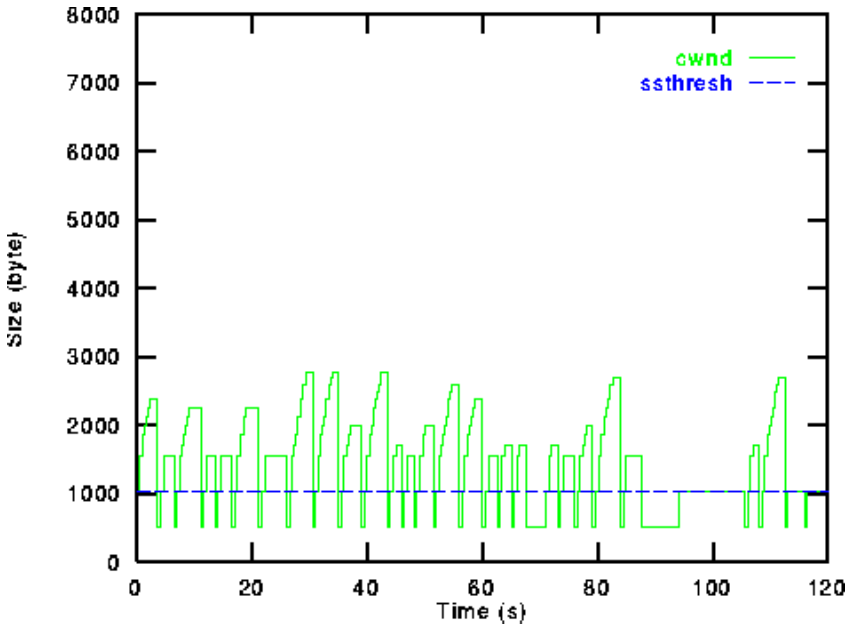
- Analoge Situation, jetzt kommt es aber zu Wiederholungen von 30% aller TCP-Datensegmente
  - z.B. aufgrund von typischen Überlastsituationen im Internet
  - nach 120 Sekunden sind diesmal erst ca. 140000 Byte übertragen



- Durchsatz nur noch ca. 10 kb/s; stark schwanken:

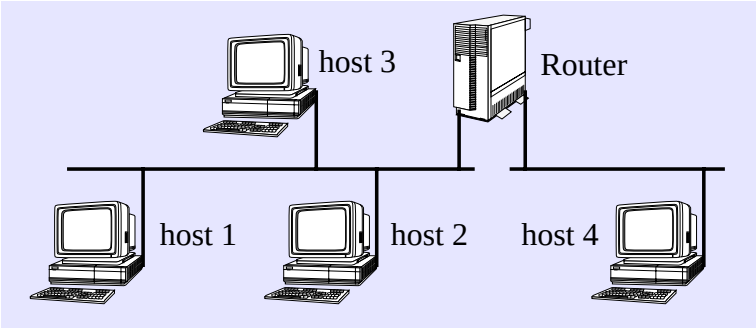


- Congestion Window kann kaum wachsen:

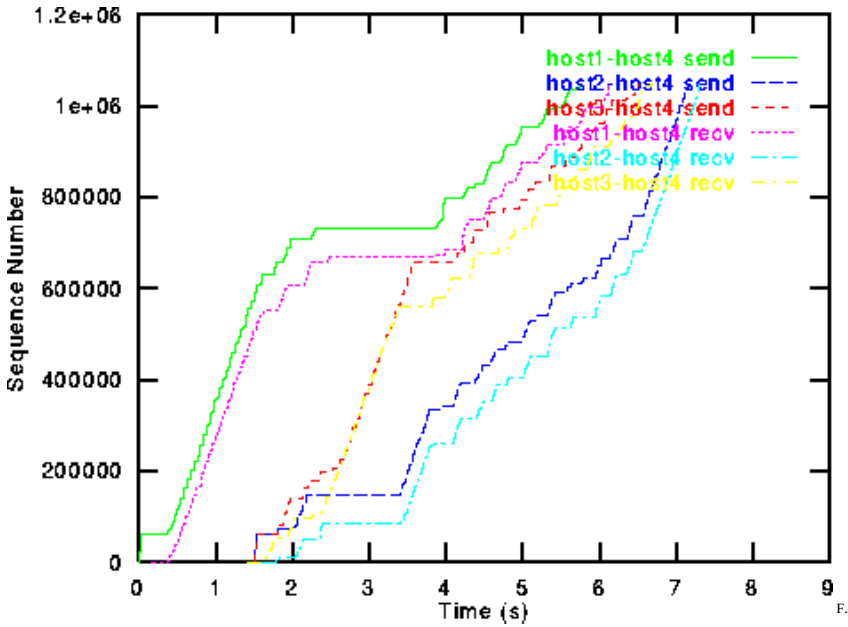


TCP-Benchmark (4)

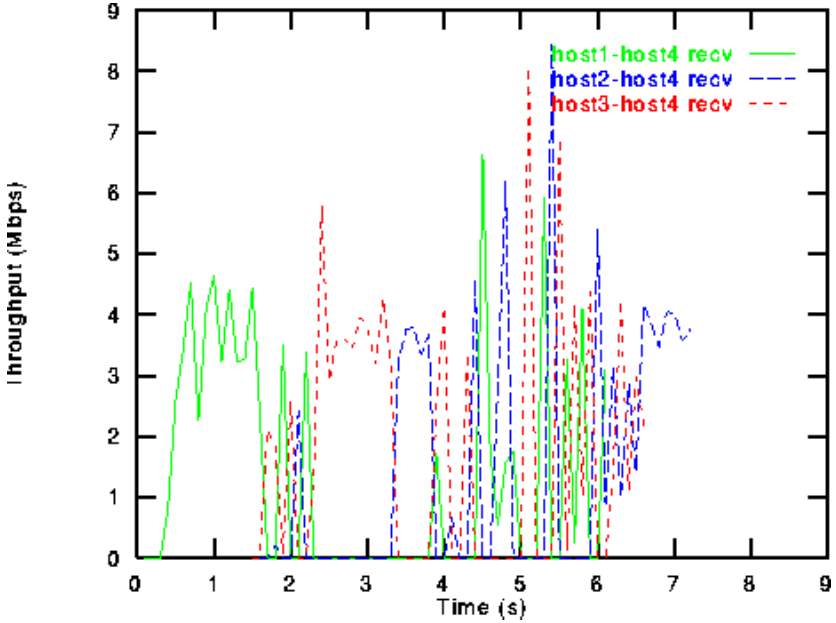
- 2 Ethernet-Segmente verbunden durch einen Router
  - LAN aus klassischem Ethernet mit 10 Mb/s



- Host 1, host 2 und host 3 senden jeweils 1000 Datenpakete zu 1024 Byte an host 4
  - host 2 und host 3 starten den Datentransfer zeitversetzt um 1 Sekunde



- Durchsatz:



- Beobachtungen:
  - Durchsatz ist selten höher als 5 Mb/s; im Mittel ca. 3.7 Mb/s
  - zeitweise Monopolisierung (z.B. von 2.5 bis 3.5 durch host 3)

- Denkübung: wie könnte man die Phänomene erklären?
  - hier spielt u.a. das Ethernet-Protokoll (CSMA/CD) und das TCP-Protokoll hinein!

Adressierung im Internet

- Zusammengefasst ergibt sich folgendes Bild:

Application Layer	- symbolischer Domain-Name, z.B. in einer URL bei http
Transport Layer (TCP, UDP)	- Port-Nummer (2 Byte)
Network Layer (IP)	- IP-Adresse (4 Byte)
Link Layer Physical Layer	- z.B. Ethernet-Adresse (6 Byte)

- Hierbei sind die Pakete i.a. ineinander verschachtelt!

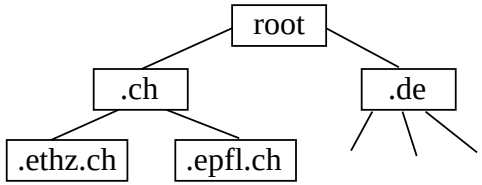
Namen und Adressen

- *Namen* geben i.a. Aufschluss über die *Art* eines Objektes
  - Typ, Gestalt, Zweck... (falls Name sinnvoll gewählt!)
  - “Namen sind Schall und Rauch” oder “Nomen est Omen”?
- Namen dienen auch der *Bezeichnung* und *Identifizierung*
  - daher oft auch “Bezeichner” oder “Identifikator” für “Name”
  - es gibt auch *anonyme* Objekte (z.B. dynamisch erzeugte Variablen)
  - ein Objekt kann u.U. *mehrere Namen* haben (“alias”)
  - gleicher Name kann in verschiedenen Kontexten (“Namensraum”) unterschiedliche Objekte bezeichnen
- *Adresse* ermöglicht die *Lokalisierung* eines Objektes
- Adressen sind innerhalb eines Kontextes (“Adressraum”) eindeutig
- Adresse eines Objektes ist u.U. *zeitabhängig*
  - mobile Objekte
  - “relocatable”
- *Dagegen*: Name eines Objektes ändert sich i.a. nicht
  - vgl. aber: Namensänderung bei Heirat, Zuweisung eines Alias...!
- Entkoppelung von Namen und Adressen unterstützt die *Ortstransparenz*
- Daher dynamische Zuordnung Name --> Adresse nötig
  - vgl. persönliches Adressbuch
  - “Binden” eines Namens an eine Adresse

Internet Domain Name System (DNS)

- Jeder Rechner im Internet hat eine IP-Adresse
  - typischerweise als 4 Dezimalzahlen geschrieben (z.B. 192.130.10.121)
- Symbolische Namen sind besser
  - z.B. Domain-Namen wie www.nanocomp.inf.ethz.ch
  - gut zu merken; relativ unabhängig von spezifischer Maschine
  - muss bei *vor* Verwendung bei Internet-Diensten (telnet, ftp, Email, WWW...) in eine IP-Adresse umgesetzt werden
  - Umsetzung in IP-Adresse geschieht im Internet mit DNS
- Domains
  - hierarchischer Namensraum der symbolischen Namen im Internet
  - "Toplevel domains" ch, de, fr, nl (ISO 3166 Ländercodes); edu, com,...
  - Domains (ggf. rekursiv) gegliedert in Subdomains

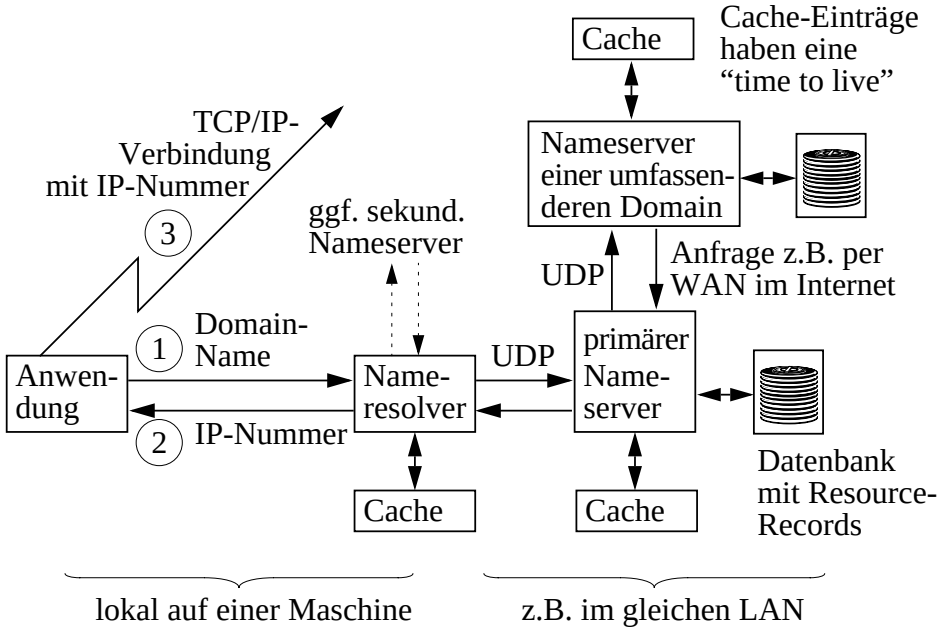
ch
ethz.ch
inf.ethz.ch
nanocomp.inf.ethz.ch
sun6.nanocomp.inf.ethz.ch



- Für einzelne Subdomains bzw. einer Zusammenfassung einiger Subdomains (sogenannte "Zonen") ist jeweils ein Domain-Nameserver zuständig
  - primärer Nameserver
  - optional zusätzlich einige weitere sekundäre Nameserver
  - oft sind Primärserver verschiedener Zonen gleichzeitig wechselseitig Sekundärserver für die anderen
  - Nameserver haben also nur eine Teilsicht!

Namensauflösung im Internet

- Historisch: Jeder Rechner hatte eine Datei hosts.txt, die jede Nacht von zentraler Stelle aus verteilt wurde
- Jetzt: lokaler Nameresolver, der eine Zuordnungsdatei /etc/hosts für die wichtigsten Rechner enthält, und sich ansonsten an einen seiner nächsten Nameserver wendet
  - IP-Nummern der lokalen Nameserver stehen in der Datei resolv.conf



nslookup

NAME in.named, named

in.named is the Internet domain name server. It is used by hosts on the Internet to provide access to the Internet distributed naming database. See RFC 1034 and RFC 1035 for more details. With no arguments, in.named reads /etc/named.boot for any initial data, and listens for queries on a privileged port.

NAME nslookup - query name servers interactively

nslookup is an interactive program to query Internet domain name servers. The user can contact servers to request information about a specific host, or print a list of hosts in the domain.

```
> sun20
Name:    sun20.nanocomp.inf.ethz.ch
Address:  129.132.33.79
Aliases:  ftp.nanocomp.inf.ethz.ch

> altavista.com
Name:    altavista.com
Addresses:  204.123.2.75, 204.123.2.66,
204.123.2.69

> altavista.com
Name:    altavista.com
Addresses: 204.123.2.66, 204.123.2.69,
204.123.2.75

> cs.uni-sb.de
Name:    cs.uni-sb.de
Addresses:  134.96.254.254, 134.96.252.31
```

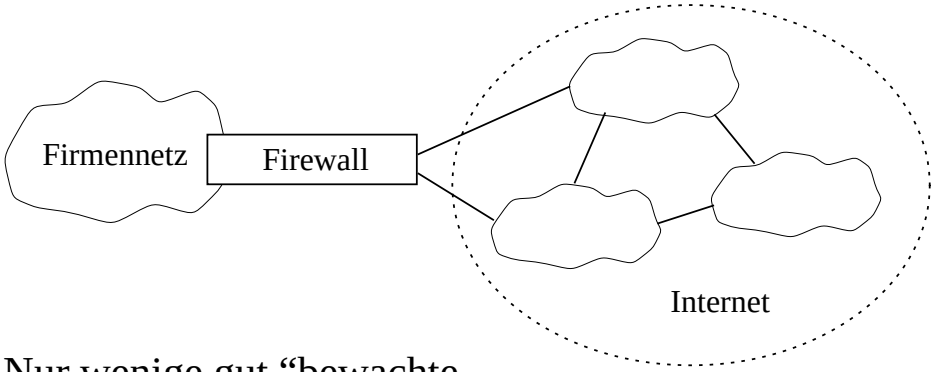
Dies deutet auf einen "round robin"-Eintrag hin: Der Nameserver von altavista.com ändert alle paar Minuten die Reihenfolge der Einträge, die bei anderen Nameservern auch nur einige Minuten lang gespeichert bleiben dürfen. Da Anwendungen i.a. den ersten Eintrag nehmen, wird so eine Lastverteilung auf mehrere Altavista-Server vorgenommen!

Router an zwei Netzen

Intranet

- Nutzung der Internet-Technologien und -Konzepte für firmeninterne Netze
  - TCP/IP-Protokoll (+ darauf aufbauende Dienste)
  - Web-Browser (+ http, html-Dokumente...)
  - ==> einheitliche Verfahren und Oberflächen, um von jedem Arbeitsplatz (PC, MAC, Workstation) aus an die Firmendaten zu kommen
- Aufgabe: Server und Clients "intranetfähig" machen, z.B.:
  - Java-Applets als lokal laufenden Anwendungen auf Client-Seite
  - Datenbank-Server mit http-Schnittstelle
- Intranet-Infrastruktur als Problem- und Geschäftsfeld
  - firmeninterne Suchmaschinen
  - Hochleistungs-Webserver
  - Sicherheitskonzepte
  - Datenbankanschluss
- Intranet ermöglicht relativ problemlose Verbindung privater Netze mit dem Internet
  - Lücke schliessen zwischen firmeneigenen Informationssystemen und dem Internet (--> electronic commerce...)
  - Sicherheitsproblematik:
    - Firmennetz als Teil des Internets?
    - Internet-Protokolle und -Dienste wurden nicht für sicherheitskritische Anwendungen entworfen!
    - Zauberwort "Firewall"

Firewall (1)

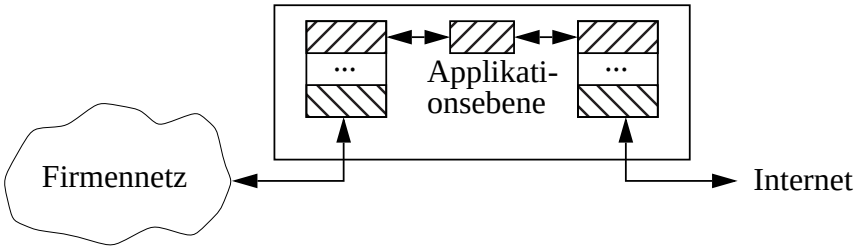


- Nur wenige gut “bewachte Übergangsstellen” in das Internet bereitstellen
  - vgl. Zugbrücke einer mittelalterlichen Burg
  - Überprüfen von Datenpaketen, Verbindungswünschen, Adressen...
  - Verhindern gewisser Kommunikationsmöglichkeiten
  - Protokollierung, Alarmmeldungen etc.
  - Beachte: Eine Firewall muss gut administriert werden!

- Typ “Paketfilter” (auf IP- und TCP-Ebene)
  - typischerweise spezielle Router
  - “Screening” durch definierbare Regeln
  - Überprüfen von Quell- und Zieladresse
  - Überprüfen von Port-Nummern
    - z.B. Telnet: 23; Finger: 79 etc.
    - dadurch z.B. verhindern, dass Telnet-Verbindungen von ausserhalb auf Firmenrechner möglich sind
  - transparent für die Anwender (ausser bei “verbotener” Kommunikation!)
    - Nachteil: Unterscheidung zwischen Nutzern mit verschiedenen Rechten auf dieser Ebene kaum möglich
    - Nachteil: Es gibt Dienste mit dyn. Portnummern; es gibt sogar dynamische IP-Adressen...

Firewall (2)

- Typ “Application Gateway”



- Rechner mit zwei Netzzugängen (“dual-homed”), wobei das “IP-Forwarding” dazwischen deaktiviert ist
- Für spezielle Dienste (z.B. telnet, email, ftp, WWW...) sind im Gateway Stellvertreter-Services (“Proxies”) realisiert
  - sowohl Client als auch Server wenden sich an den Proxy
  - direkte Kommunikation zwischen Client und Server ist unterbunden
  - Proxy sollte für Anwender transparent sein
  - Proxy-Software kann konfiguriert werden; damit z.B.:
    - Überprüfen von Adressen etc
    - Prüfen von Legitimation und Autorisierung
    - Authentifizierung, Einmalpasswörter, Verschlüsselung...
    - detailierte Protokollierung
    - verbergen der firmeninternen Adressen, Netzstruktur etc.

- Es gibt weitere Sicherheitstechniken in Firewalls
  - z.B. “Policy Routing” (Wahl der Routen abhängig vom Absender etc.)