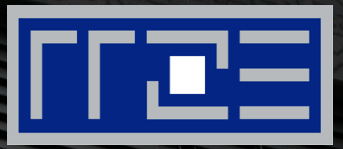
REGIONALES RECHENZENTRUM ERLANGEN [RRZE]



Traffic Engineering

Netzwerkausbildung – Praxis der Datenkommunikation

20.01.2016, Jochen Reinwand, RRZE

Gliederung

- Begriffserklärung und Motivation
- Techniken des Traffic Engineering
  - Layer 2
  - Layer 3
  - Layer 4
  - Layer 7
- Zusammenfassung

MOTIVATION

Begriffserklärung und Motivation

Begriffserklärung

Traffic Engineering

Traffic Engineering (TE) beschreibt im Deutschen den Prozess, der die Erhebung (Analyse), Gestaltung und Optimierung von Datenflüssen und -wegen in Rechnernetzwerken zum Inhalt hat.

…

Quelle: wikipedia.de

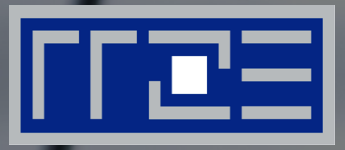
Wozu Traffic Engineering

- Beispiele für mögliche Anforderungen gegenüber TE:
  - Optimierung der Verbindung zwischen den Kommunikationspartnern
  - Erhöhung der Dienstgüte
  - Lösen von komplexen Problemen im Bereich des Netzwerkdesigns/Routings
  - Erhöhung der Sicherheit

Wozu Traffic Engineering

- Beispiele für Maßnahmen
  - Bandbreitenmanagement bei überlasteten Strecken
  - Definieren von VPNs/Tunneln/virtuellen Standleitungen,…
  - Verbesserung der Latenz für zeitkritische Netzdienste
  - Einführung eines Lastausgleichssystems auf bestimmte Dienste (z.B. load-balanced Webserver)
  - Begrenzung des anfallenden Datenverkehrs
- Spannungsfeld Netzneutralität
  - *„Middleboxen verkalken das Internet”*
    http://heise.de/-2133877

TECHNIKEN DES TRAFFIC ENGINEERING



Traffic Engineering auf Layer 2

Layer 2 – Ansatzpunkte

- „Virtualisierung“ der physikalischen Struktur
  z.B.: Ethernet VLANs, ATM PVCs (Asynchronous Transfer Mode Permanent Virtual Circuit)
- Optimierung des Versands der Datenpakete (QoS/CoS)
- Load-Balancing über mehrere physikalische Strecken („Channel-Bundling“)

→ Möglichkeiten für TE stark abhängig von der verwendeten Netztechnologie (ATM? LAN?? WLAN???)

Beispiel ATM

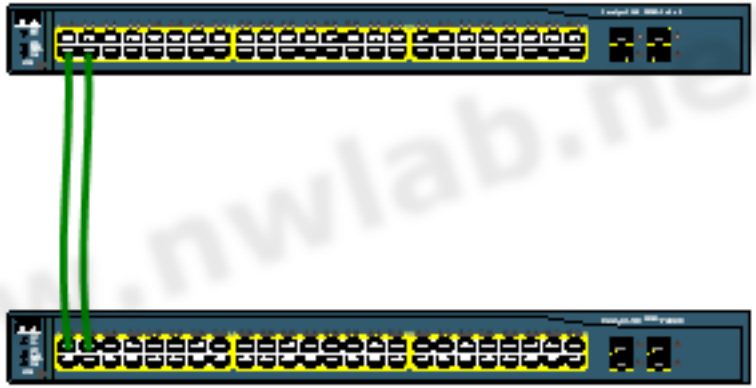
- *Virtuelle* Verbindungen
  - Signalisierung und Verbindungsaufbau entlang eines festgelegten Pfades
  - Datenframes („Zellen“) mit fester Größe
- Sehr gute TE-Möglichkeiten im Bereich QoS
  - Unterschiedliche QoS-Klassen pro Verbindung: *Constant Bitrate, Variable Bitrate, Unspecified Bitrate, Available Bitrate*
  - Praktisch überall im Rückbau (Ersatz durch Ethernet, IP, VPN, MPLS)
- Weitere TE-Anwendung: „Ethernet-over-ATM“
  - Nicht trivial (Ethernet Verbindungslos vs. ATM verbindungsorientiert)
  - Emulation eines LANs mittels sog. LAN-Emulation („LANE“)

Beispiel Ethernet

- Nur beschränkte TE-Möglichkeiten bei IEEE 802 Ethernet
- Virtualisierung der Netze möglich mittels VLAN-Konzept (statisch oder dynamisch)
- CoS bzw. „QoS-Light“: 802.1p/q
  - Class of Service ↔ Quality of Service
  - 3 Bit-Feld im Ethernet-Frame
  - Grobe Klassifizierung von Ports/VLANs/MACs möglich
  - Definition z.B. von Priorität, Bandbreite, Verkehrsvolumen

Beispiel Ethernet

- Ethernet Channel-Bundling: Zusammenfassung mehrerer Ethernet-Links zu einem logischen Ethernet-Link („Etherchannel“, „Bonding“, „Trunking“, „Teaming“,…)
- Standardisiert als 802.3ad bzw. “ LACP“
- Ethernet-Frames werden auf vorhandene Links verteilt
- Aufteilung konfigurierbar:
  - Round-Robin
    (Gefahr von out-of-order delivery)
  - Hashing-Verfahren
    (Quell- und/oder Ziel-MAC)
  - Weitere: adaptiv, dynamisch



Quelle: http://www.nwlab.net

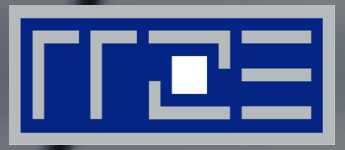
Beispiel Wireless-LAN (WLAN)

- Gemeinsam verwendetes, unzuverlässiges Medium „Funk“
- Verwaltung des Medienzugriffs: Dezentral durch CSMA/CA
- Störungen/Empfangsbedingungen kaum vorhersehbar!
- TE im Sinne von QoS schwierig bis unmöglich
- Verbesserungsansatz:
  - *„hybrid coordination function“* (IEEE 802.11e; verschiedene Prioritäten; Access-Point als Koordinator)
  - Aber: Physikalisches Grundproblem (Störungen) bleibt bestehen
- VLAN-Konzept bei WLAN schlecht durchführbar

Zusammenfassung Layer 2

- Möglichkeiten für TE vor allem abhängig von jeweiligen QoS-Funktionalitäten:
  - ATM: relativ gut
  - Ethernet-LAN: mäßig
  - Ethernet-WLAN: schlecht
- TE durch virtuelle Netzstrukturierung
  - Ethernet (VLANs)
  - ATM (LANE)
  - Bei WLAN prinzipbedingt schlecht möglich.
- ATM auf dem Rückzug (Quantität schlägt Qualität)

TECHNIKEN DES TRAFFIC ENGINEERING



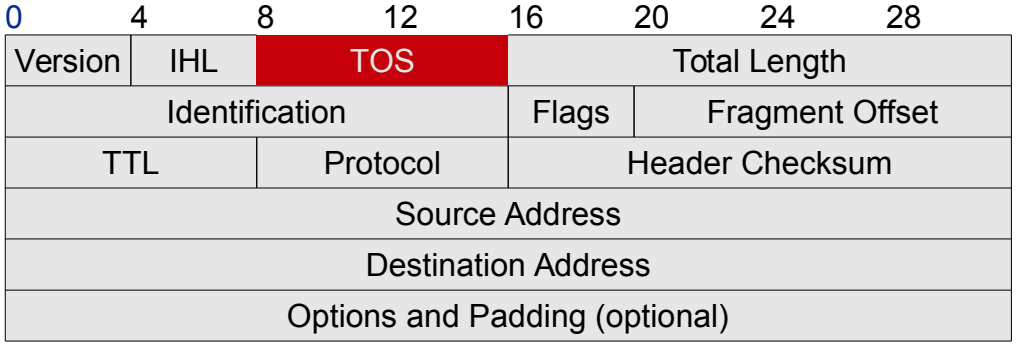
Traffic Engineering auf Layer 3

Layer 3 – Ansatzpunkte

- Priorisierung von IP-Paketen
- Beeinflussung des Routings der Pakete
  - Destination-Based-Routing
  - Source-Based-Routing
  - Policy-Based-Routing
- IP Load-Balancing
- Label-Switching statt Routing: MPLS

Priorisierung von IP-Paketen

- Type of Service (TOS)
  - 8-Bit Feld für Prioritätsklassen (RFC 791,1349,2474,…)

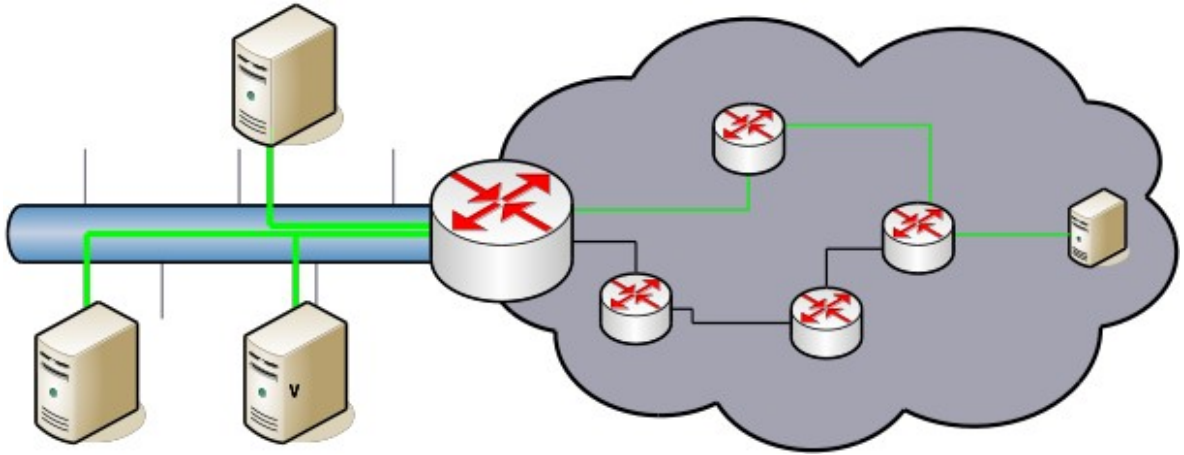


- Bsp. DiffServ (RFC 2474): Differentiated Services
  - Nutzung von 6 Bits des TOS-Felds: 3 Bits für Klasse, 3 Bits für „Drop Precedence“
  - Bsp: „000000“ = best efford, „101110“ = premium-class
  - Nur gezielte Priorisierung von Paketen möglich, d.h. CoS, kein QoS
  - Kann(!) von Routern ausgewertet werden
  - Definition von „Trust Boundaries“ gegen Missbrauch

Traffic Engineering und Routing

Standardfall: Destination-based-Routing

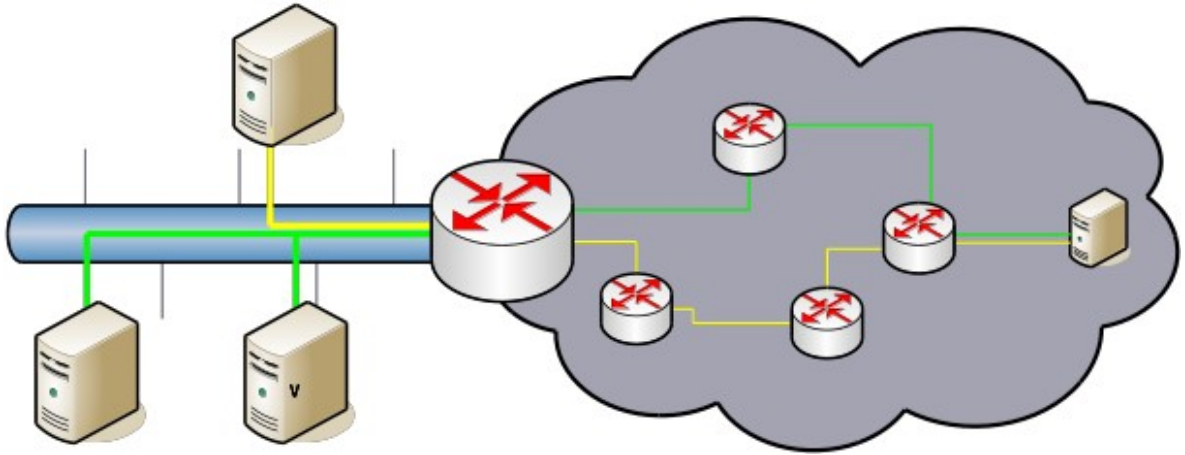
- Routing-Entscheidung auf jedem Router basiert auf Zieladresse
- Vorhalten von Routing-Tabellen in den Routern (Routing-Entscheidung über sog. *Longest Prefix Match*)
- Dynamisches Routing (OSPF, RIP, BGP,…)
- Statisches Routing



Traffic Engineering und Routing

Sonderfall: Source-based-Routing

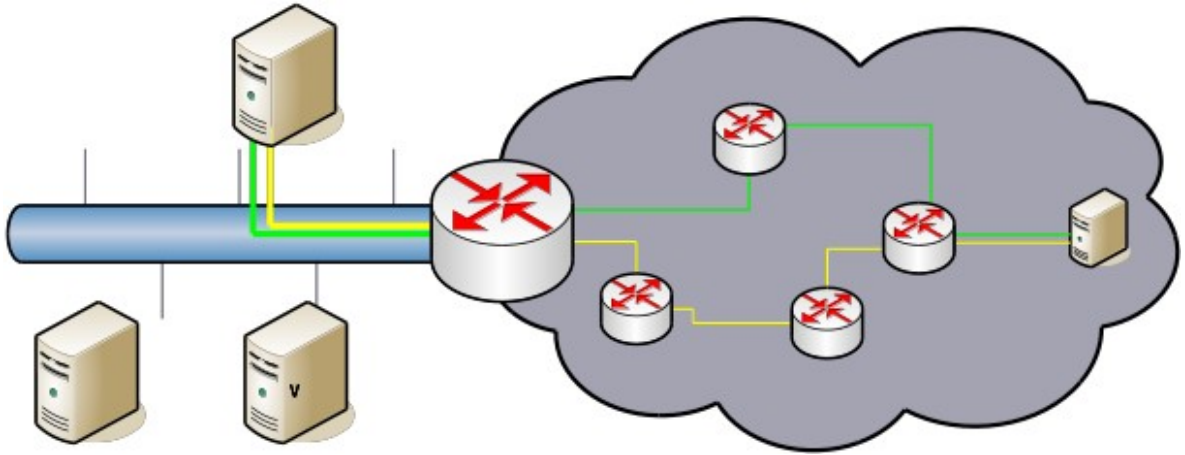
- Routing-Entscheidung basiert auf Absenderadresse
- Pakete verschiedener Sender im gleichen Netz können verschiedene Wege zum gleichen Ziel nehmen
- Bsp: Verbindungen eines bestimmten Rechners im Subnetz sollen über eigene Leitung geroutet werden
- Problem: Rückroute! (i.d.R. normales dest.-based Routing!)



Traffic Engineering und Routing

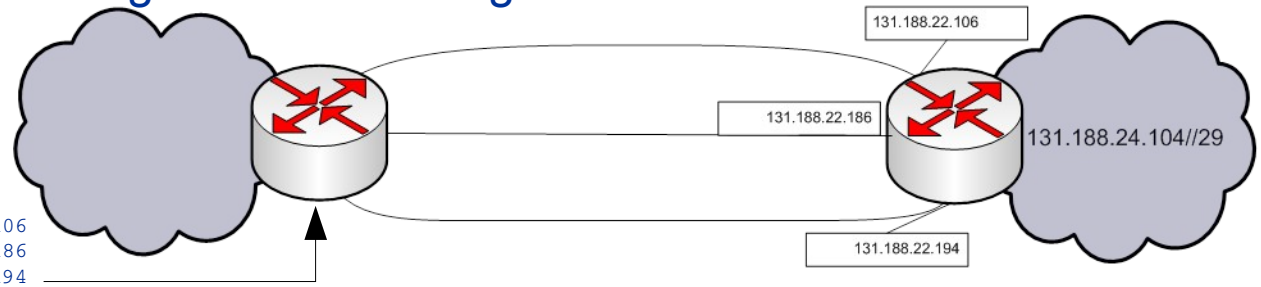
Allgemeiner Fall: Policy-based-Routing

- Routing-Entscheidungen über verschiedenste Kriterien (z.B. IP-Flags, Höhere Protokolle,..)
- Bsp: Bestimmte Protokolle (z.B. ssh) sollen über eine besonders Latenzarme Leitung geroutet werden
- Problem: Rückroute



IP Load-Balancing

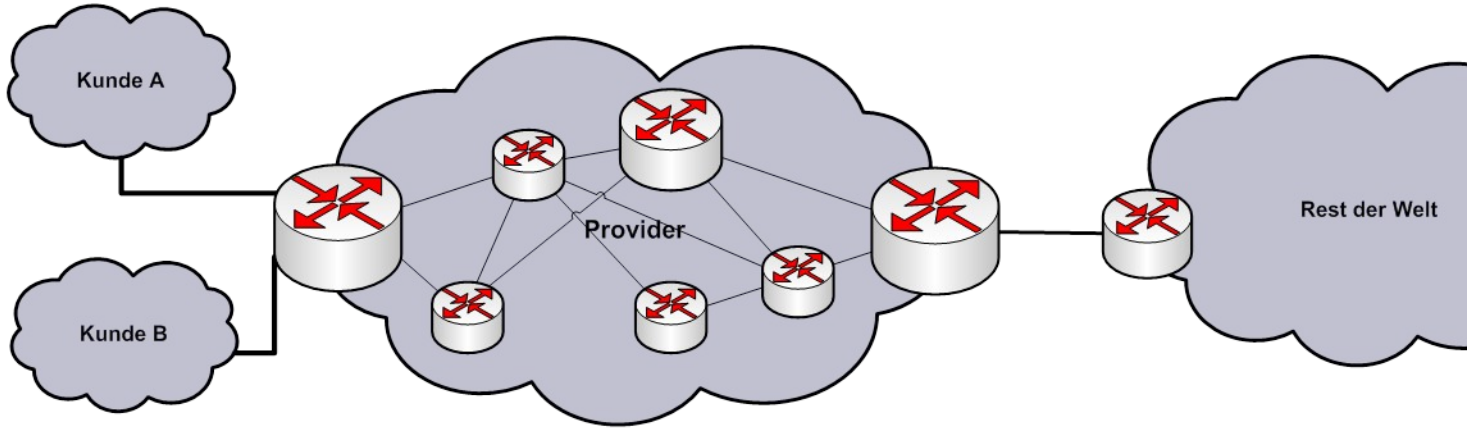
- Prinzip wie L2-Load-Balancing (Channel-Bundling)
- Jedoch: Aufteilung von IP-Paketen statt Frames
- Aufteilung der Pakete auf Leitungen nach
  - Verbindungen (d.h. „Network-Flows“: für Router einfach, aber oft nicht ideal)
  - Round-Robin (optimale Auslastung, aber Problematik der „Out-of-Order“ Delivery)
- Implementationsabhängig!
- Einfaches Load-Balancing möglich durch Konfiguration von mehreren statischen Routen für gleiches Ziel mit gleicher Metrik:




```
ip route 131.188.24.104/29 131.188.22.106
ip route 131.188.24.104/29 131.188.22.186
ip route 131.188.24.104/29 131.188.22.194
```

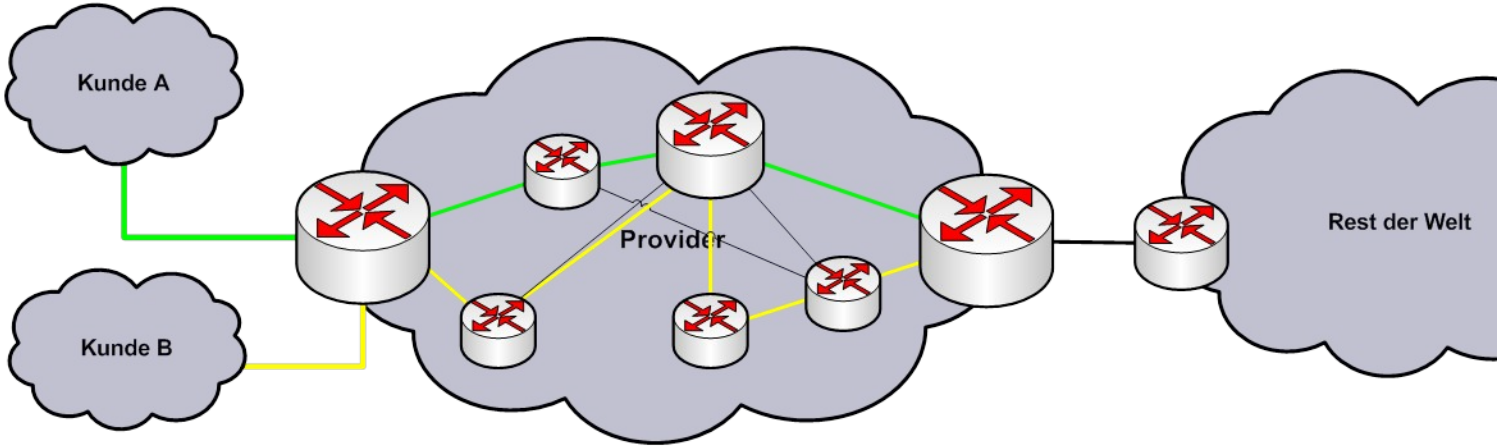
MPLS – Motivation

- Provider besitzen große, weitverzweigte IP-Backbones
- Standarddienstleistung: „Durchleitung“ von Daten durch den Backbone (z.B. Standortvernetzungen, Festanbindung ins Internet,..)
- Kunde will i.d.R. gewisse Zusagen bzgl. QoS, Bandbreite,…
- Leitungsorientierter Wunsch des Kunden
  ↔ packetorientierte Realität des Backbone



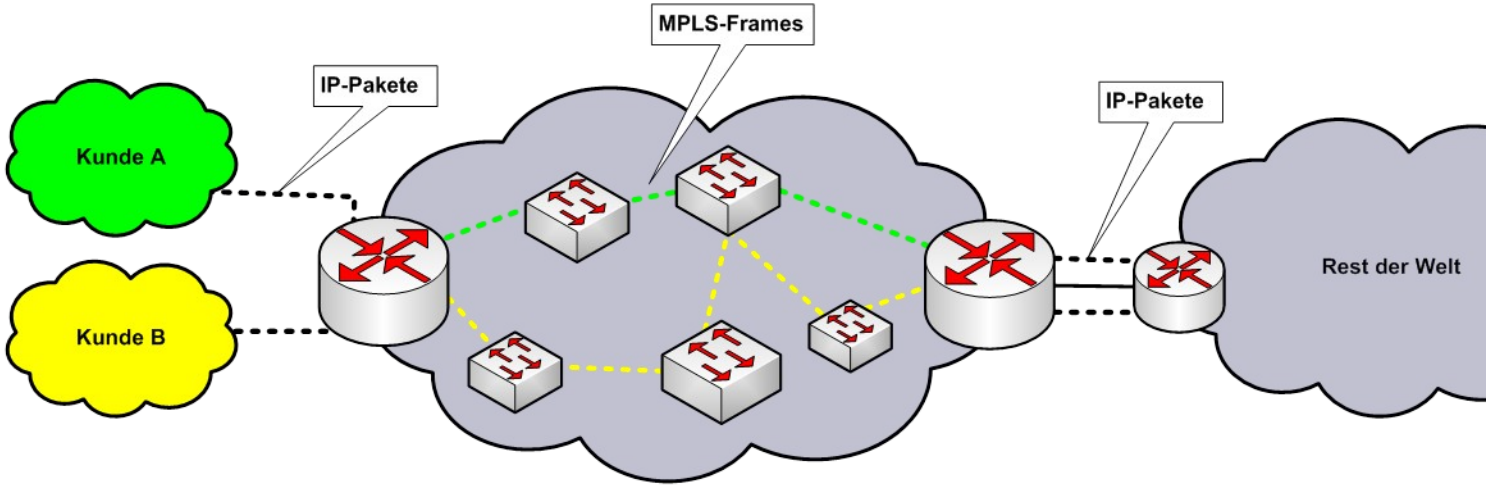
MPLS – Lösung

- *Multi-Protocol-Label-Switching*
- Idee: Packetvermittelnde Backbone-Router erweitert um Fähigkeiten der Leitungsvermittlung
- Pfad der Pakete wird im Voraus konfiguriert
- Pakete auf Pfad geforwardet, nicht einzeln von Hop zu Hop („Switching“ statt „Routing“)
- „Festverschalteter Datenpfad“ pro Kunde mit garantierten Ressourcen



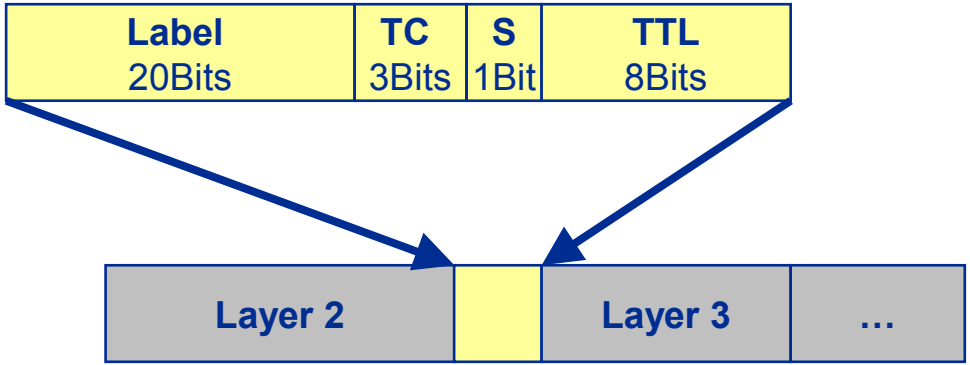
MPLS – Lösung

- Prinzip: IP-Pakete an *MPLS-Borderrouter* mit *MPLS-Label* markiert → *MPLS-Frame*
- Signalisierung des kompletten Pfads im Vorfeld
  - Aufbau von *Label-Tabellen* auf Routern
  - Ressourcenreservierung entlang des Pfades
- Pakete ohne Label werden weiterhin normal geroutet



MPLS – Technische Sicht

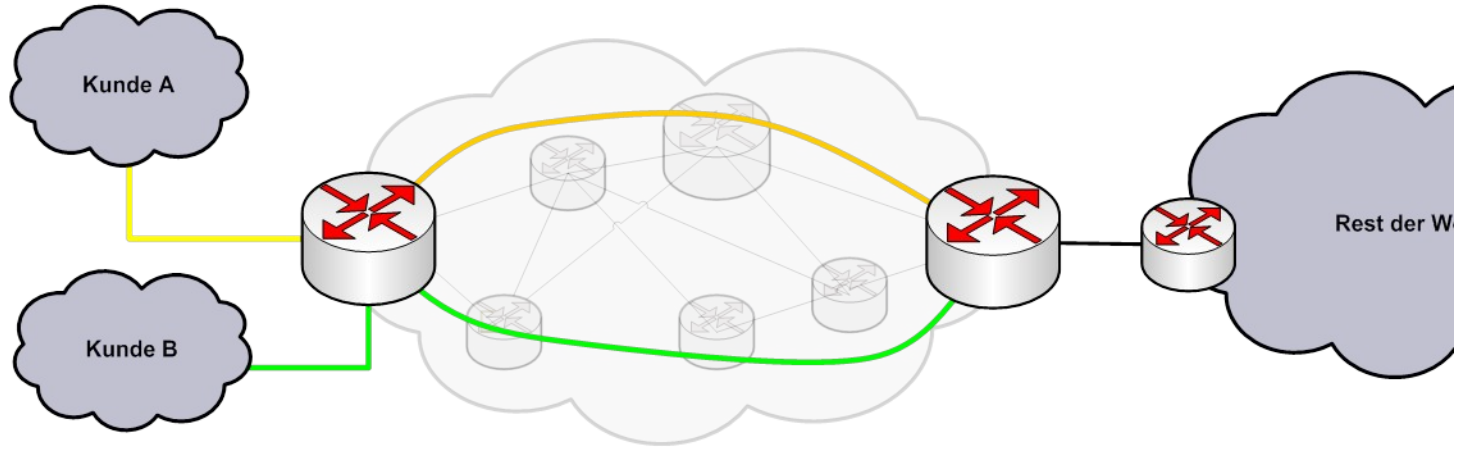
- MPLS-Eingangsrouter fügt MPLS-Header in das zu switchende Paket ein (*„Label push“*)
- MPLS-Ausgangsrouter entfernt MPLS-Header wieder (*„Label pop“*)
- MPLS als *„Layer 2.5“*



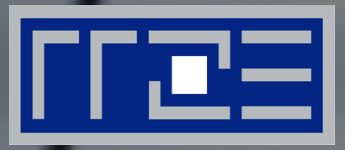
Label: 20-Bit Label Kennzeichner (lokal pro Router gültig!)
TC: Traffic Class: CoS Kennzeichner
S: Bottom of Stack: Keine weiteren MPLS-Header folgen
TTL: Hop-Counter

MPLS – Vorteile

- MPLS-Verbund *transparent* für das restliche IP-Netz (Kunde/Internet)
- Label-Switching *schnell* und *latenzarm*
- Bessere *QoS-Funktionalität* und *Determinismus* als IP-Routing
- *„Tunnel-Charakter“* → Ideal für VPN-Kopplungen oder „virtual leased line“
- *„Multiprotocol“* Technik: Beliebige Protokolle transportierbar (nicht nur IP)



TECHNIKEN DES TRAFFIC ENGINEERING



Traffic Engineering auf Layer 4

Layer 4 – Ansatzpunkte

- Einbeziehung der Informationen des Layer 4
  - Portnummern
  - Eigenschaften der Protokolle (Bsp „virtuelle Verbindungen”)
- Beispiele:
  - (Firewalling/ACL)
  - (Policy-Based-Routing)
  - Traffic-Shaping
  - Network Address Translation (NAT)

Traffic Shaping

- Optimierung des Datenflusses der Pakete beim Senden
- Einsatz sinnvollerweise auf Router
- Unidirektionales Verfahren (ohne Mitarbeit der Gegenseite; Router kann nur effektiv regeln, was er aufs Netz schickt, nicht was er empfängt; nur indirekt über TCP-Flusskontrolle möglich)
- Meist Anwendung in Bezug auf Layer 4
- Beispiele:
  - Bevorzugung bestimmter Protokolle auf ausgelasteten Strecken (z.B. SSH als „interaktiver Verkehr“)
  - Bessere Auslastung von TCP-Datentransfers durch Bevorzugung von abzusendenden ACK-Paketen (TCP-Ack-Priorisierung)
  - Drosseln der Datenrate des Senders (Bandbreitenmanagement)

Traffic Shaping – Bandbreitenmanagement

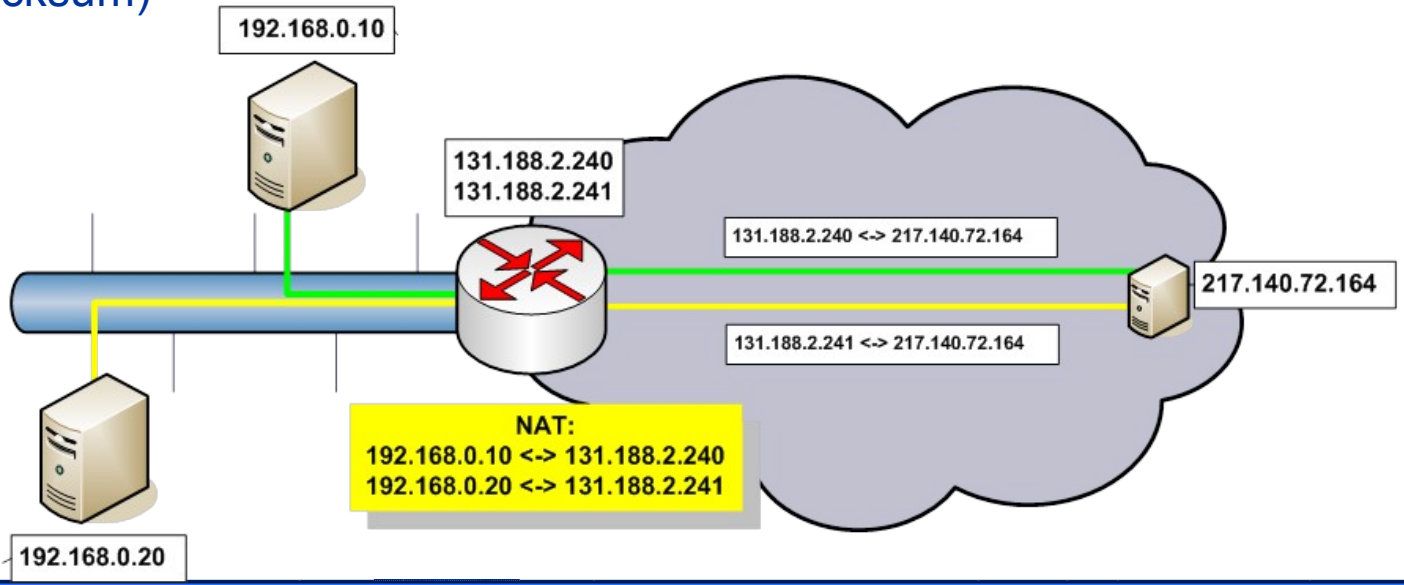
- Begrenzung der Datenrate, mit der ein Gerät bzw. Router sendet
- Bekannteste Algorithmen für Bandbreitenmanagement:
  - Leaky Bucket
    - › Daten verlassen First-In-First-Out-Warteschlange (FIFO) mit festgelegter Rate
    - › FIFO voll → Pakete werden verworfen
    - › Relativ einfach
    - → Absolut maximale, fixe Datenrate
  - Token Bucket
    - › Konstantes Auffüllen eines „Buckets“ mit „Token“
    - › Für jedes zu Übertragende Byte muss ein Token entnommen werden
    - › Token können schlagartig entnommen werden (Transferbursts möglich)
    - → Durchschnittlich maximale fixe Datenrate
  - http://www.nt.fh-koeln.de/fachgebiete/inf/vogt/mm/buckets/Applet.html

Network Address Translation (NAT)

- Umschreiben von Quell- oder Zieladresse beim Routing (IP-Layer)
- Unterscheidung:
  - Typen von NAT:
    - Basic NAT – 1:1 Umsetzung
    - PAT/NAPT – 1:n Umsetzung
      - Source-NAT (SNAT)
      - Destination-NAT (DNAT)
  - Kategorien von SNAT
    - Full Cone
    - Restricted Cone
    - Port Restricted Cone
    - Symmetric NAT

Basic NAT

- Reine 1:1 Umsetzung zwischen i.d.R. lokalen und externen Adressen
- Es werden nur IP-Adressen umgeschrieben (keine Ports)
- Aber: Trotzdem eventuell Nachbearbeitung von Protokollen höherer Schicht (Bsp: TCP-Checksum)

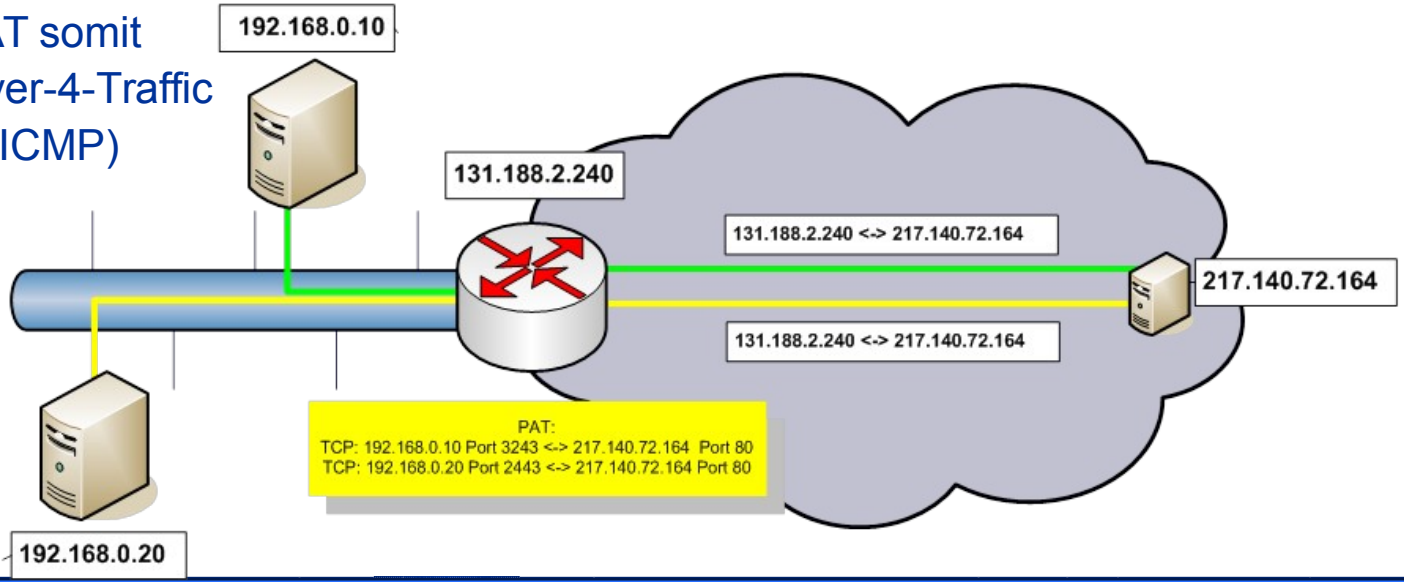


Port Address Translation (PAT)

- Auch als NAPT (Network Address Port Translation) bekannt
- 1:N Abbildungen (z.B. mehrere interne Adressen teilen sich eine externe NAT-Adresse)
- PAT Varianten:
  - Source-NAT („*Masquerading*“, N:1)
  - Destination-NAT („*Portforwarding*“, 1:N)
- Voraussetzung:
  - Nutzung von Informationen des Layer 4 (z.B. TCP/UDP-Ports)

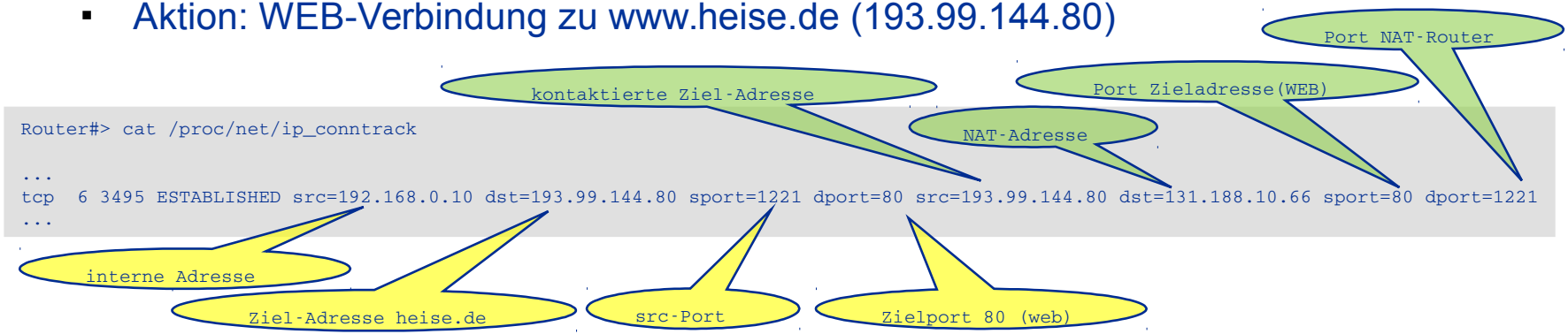
PAT – Source-NAT

- Standard-NAT-Verfahren in Home/DSL-Routern
- Bsp: Mehrere lokale Adressen gehen über eine NAT-Adresse ins Internet
- NAT-Router merkt sich pro Verbindung (d.h. Anhand TCP/UDP Ports), welche interne Adresse mit welcher externen Adresse kommuniziert
- PAT bzw. 1:N NAT somit i.d.R. nur mit Layer-4-Traffic (TCP, UDP, evtl. ICMP) realisierbar



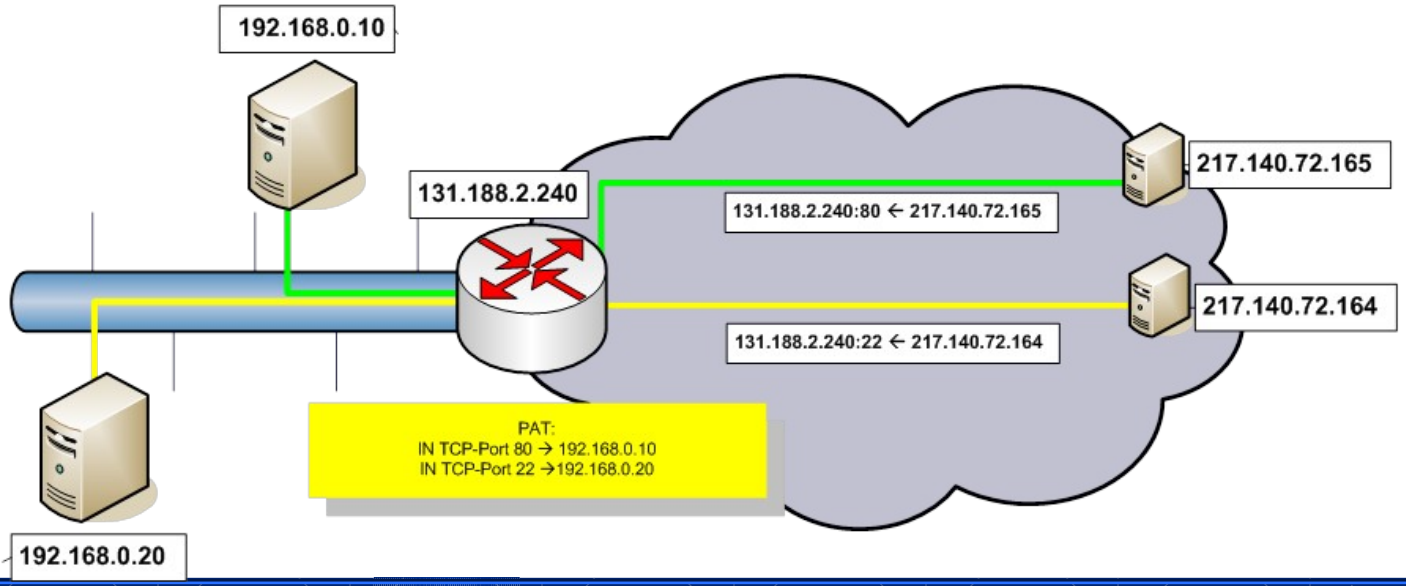
PAT – Source-NAT

- Router muss Buch führen über alle offenen Verbindungen
  → *„connection tracking table“*
- i.d.R. auf Router einsehbar (Datenschutz!)
- (Fiktives) Beispiel: NAT-Gateway auf Linux Basis
  - (Interner) Benutzer: 192.168.0.10
  - NAT-Router: 131.188.10.66
  - Aktion: WEB-Verbindung zu www.heise.de (193.99.144.80)



PAT – Destination-NAT

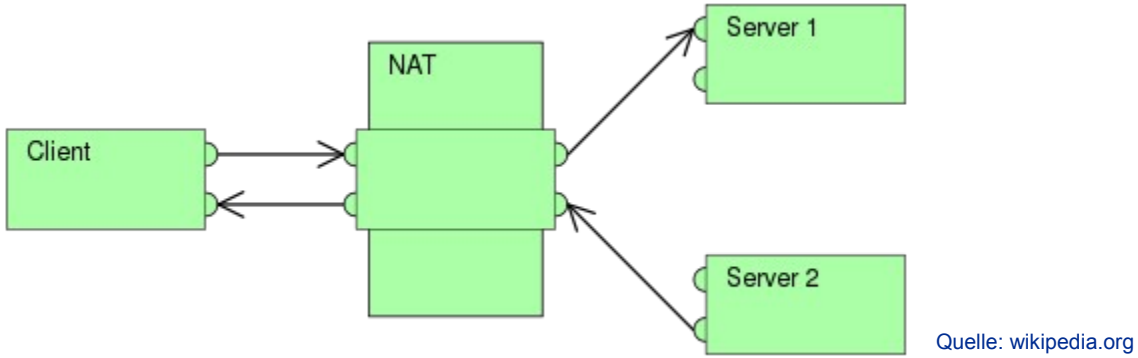
- Ebenfalls häufiger Einsatz in Home/DSL-Routern
- Für Verbindungsaufbau von Außen
- Abbildung auf interne Hosts abhängig von angesprochener Portnummer der externen NAT-Adresse



Source-NAT – Kategorisierung

- Versuch einer Kategorisierung der Varianten von SNAT
- Aufgestellt während der Entwicklung des STUN-Protokolls (Session Traversal Utilities for NAT)
- Keine sehr saubere Kategorisierung
  - div. Mischformen in der Praxis
- Vier Varianten:
  - Full Cone NAT
  - Restricted Cone NAT
  - Port Restricted Cone NAT
  - Symmetric NAT

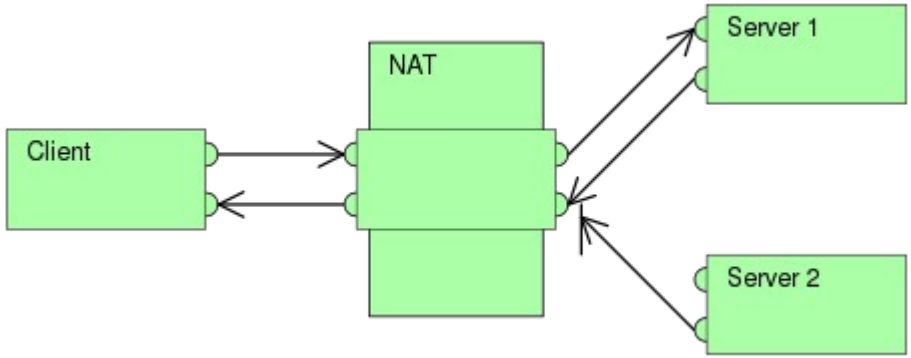
Source-NAT – Full Cone



Quelle: wikipedia.org

- Statische 1:1 Zuordnung:
  Interne Adresse/Port → NAT-Adresse/Port
- Zugriff von Außen nach Innen dann jederzeit von jeder externen Adresse möglich
- Relativ wenig verbreitet

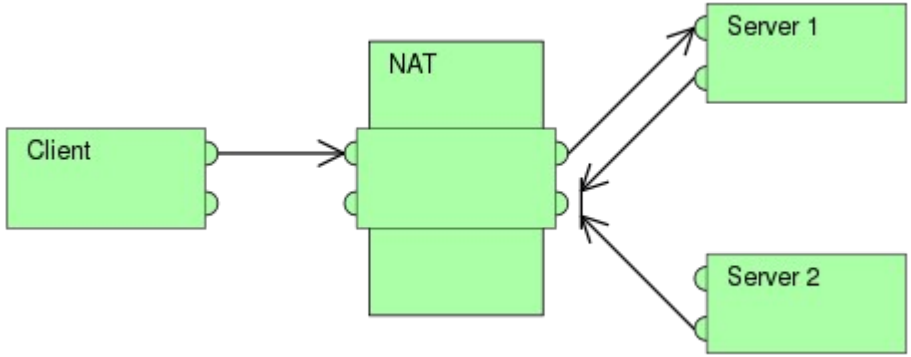
Source-NAT – Restricted Cone



Quelle: wikipedia.org

- Wie „full cone“, d.h. statische 1:1 Zuordnung:
  Interne Adresse/Port → NAT-Adresse/Port
- Aber: Zugriff von Außen nach Innen nur möglich, wenn interne Adresse bereits Kontakt mit externer hatte
- Relativ wenig verbreitet

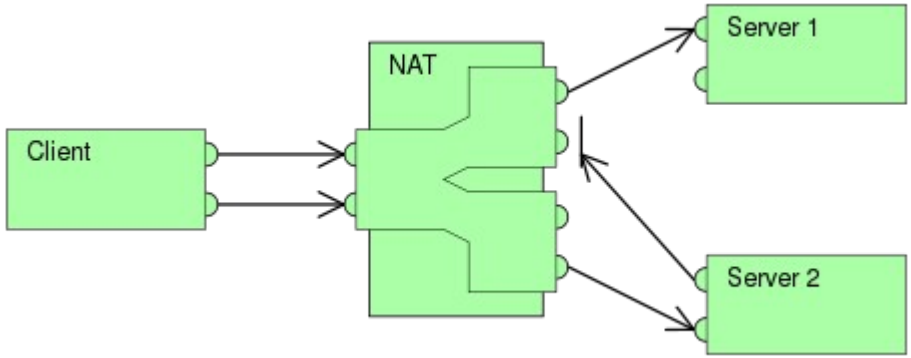
Source-NAT – Port Restricted Cone



Quelle: wikipedia.org

- Wie „restricted cone“
- Aber: Zugriff von Außen nach Innen nur möglich, wenn interne Adresse schon einmal Kontakt mit externer am jeweiligen Port hatte („statefull Verhalten“)
- In div. Varianten relativ weit verbreitet

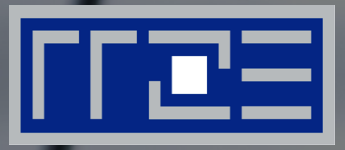
Source-NAT – Symmetric NAT



Quelle: wikipedia.org

- Mehrere NAT-Adressen (z.B. Zwecks Load-Balancing)
- Jeweils unterschiedliche Abbildung von interner Adresse/Host auf NAT-Adresse/Port (je nach externem Ziel/Port)
- Genereller Zugriff von Außen nach Innen nur schwierig
- Relativ selten

TECHNIKEN DES TRAFFIC ENGINEERING



Traffic Engineering auf Layer 7

Layer 7 – Ansatzpunkte

- Alle Arten von anwendungsbezogener Traffic-Manipulation

- Bsp:
  - Proxy-Server
  - Load-Balancer
  - Application-Level-Firewall
  - …

Proxy-Server

- Kurzform für *„Proxy representative“* (Stellvertreter)
- Proxy-Server als eine Art von Layer-7 Gateway
  - Proxy-Server meist am Netzübergang
  - Erlaubt Anwendungen in andere Netze zu kommunizieren
  - Aber kein Routing oder NAT der Pakete
- Clients teilen Proxy-Server Verbindungswunsch mit

  → Proxy baut als „Vertreter“ eigene Verbindung zum Ziel auf

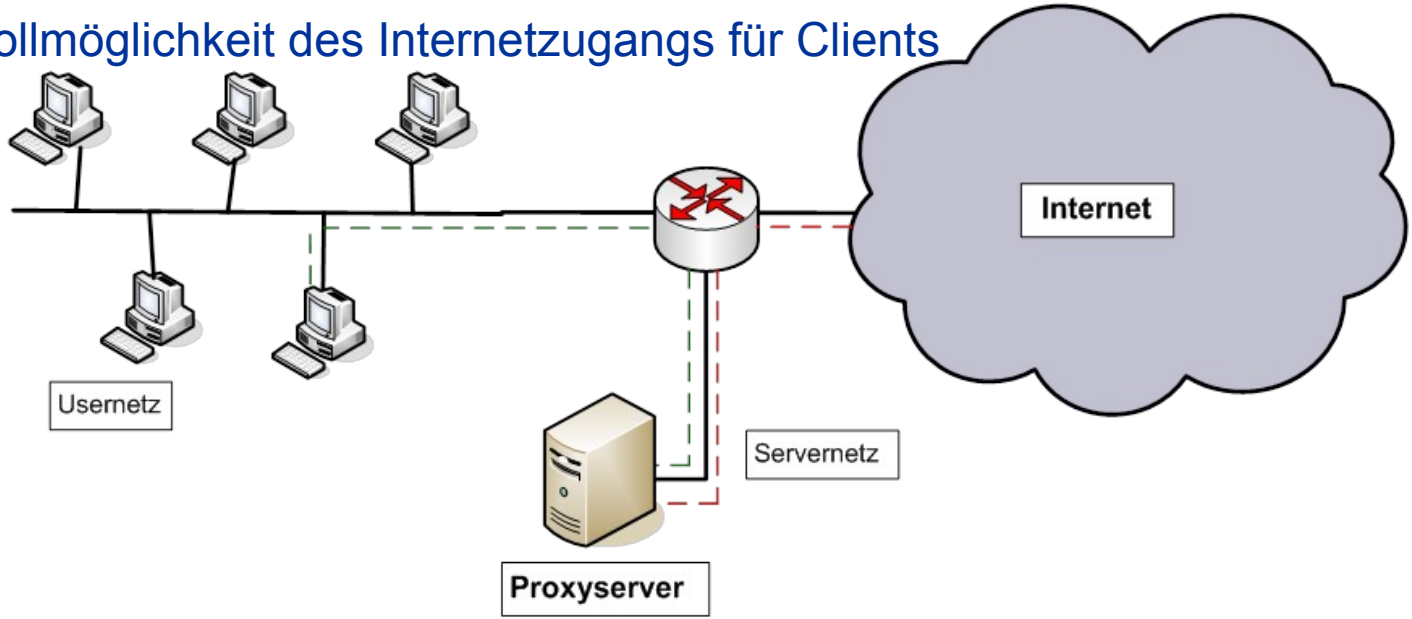
  → Übertragung der Daten über beide Verbindung
- Proxy-Sicht: Zwei separate Verbindungen (innen und außen)
- Anwendungsprogramm kennt Proxy-Server und muss mit diesem auch umgehen können
  - Ausnahme: Sogenannte *transparente Proxies*

Proxy-Server – Einsatzzweck

- Caching
  - Performance
  - Verkehrsverminderung
  - Lastverminderung
- Content-Filter (z.B. Werbeblocker)
- Sicherheit
- Anonymisierung
- Logging
- Accounting
- …

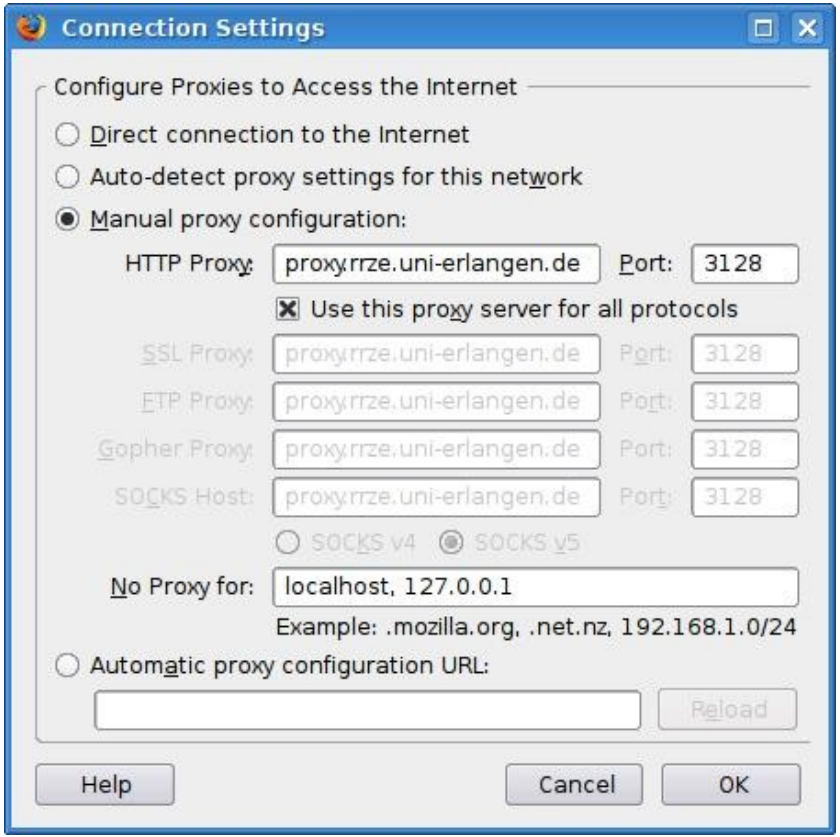
Proxy-Server – Beispiel

- Usernetz kann nicht direkt mit Internet kommunizieren (aber mit Servernetz)
- Servernetz hat vollen Internetzugang
- Server im Servernetz als Proxy z.B. für http
- Zentrale Kontrollmöglichkeit des Internetzugangs für Clients



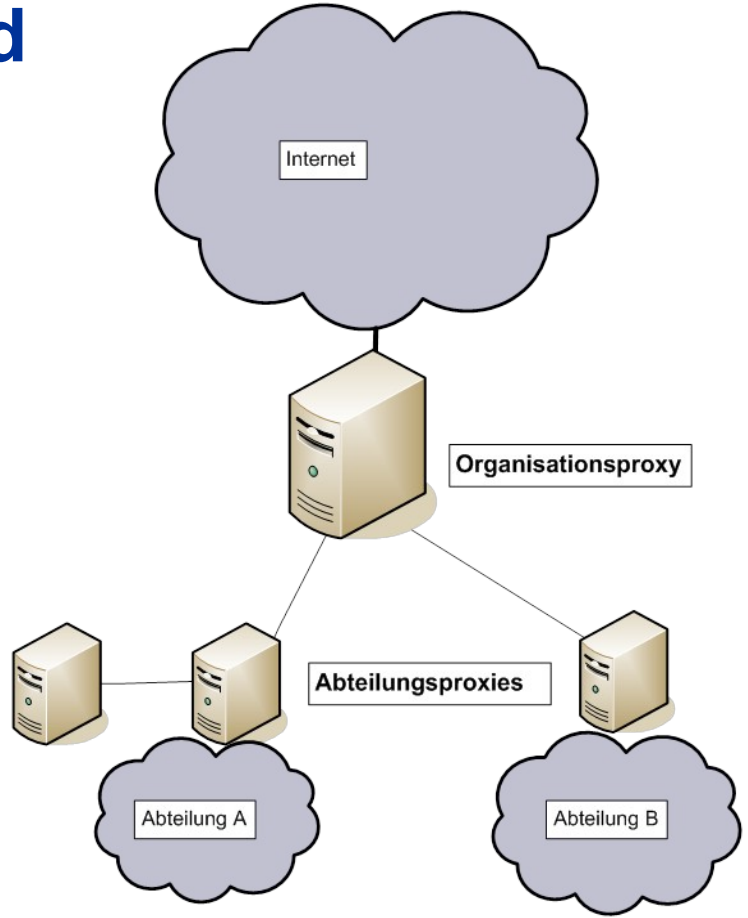
Proxy-Server – Unterstützte Protokolle

- http („squid webproxy“)
- ftp
- dns
- smtp
- telnet
- …



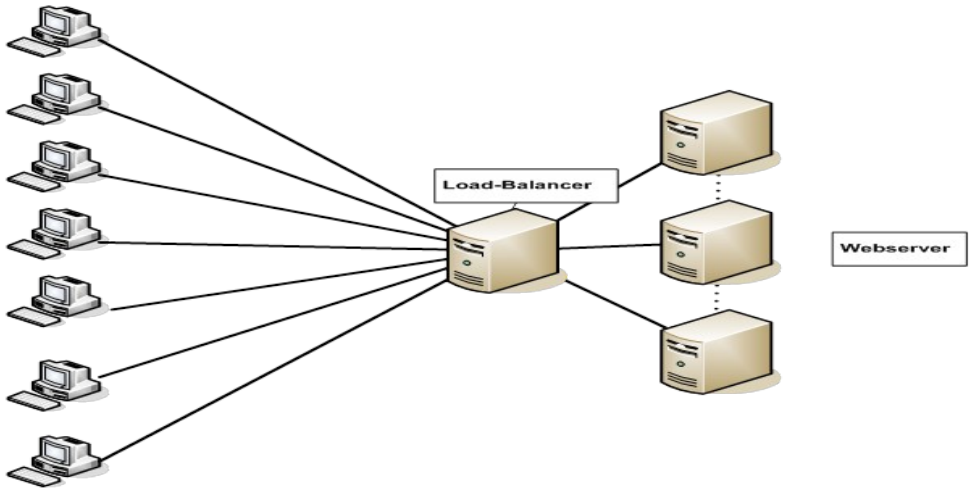
Proxy-Server – Proxy-Verbund

- Zusammenschalten/Kaskadierung von mehreren Proxies (meist http-Proxies) als gleichberechtigte Partner und/oder Hierarchien
- Ziel:
  - Höhere Performance
  - Niedrigeres Datenverkehrsvolumen nach Extern
  - Höhere Verfügbarkeit



Lastverteilung (Load-Balancing)

- Verteilung von Anfragen an einen Dienst auf mehrere gleichwertige Server
- Beispiel: Webauftritt einer großen Firma
  - Webadresse nicht bedient von Webserver, sondern von Load-Balancer (LB)
  - LB reicht Anfrage an einen von mehreren Webservern (sog. „*Real-Server*“) weiter
  - Terminierung von SSL-Verbindungen (z.B. https) auf LB möglich
  - Wenn der Load-Balancer auf Layer 7 arbeitet, kann auch eine Anfrageverteilung nach Inhalten erfolgen



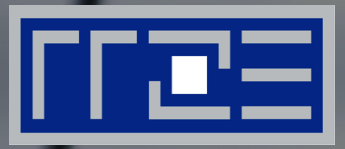
Load-Balancing – Beispiel Google

- Kombination von netz- und anwendungsbasiertem Load-Balancing
- Bsp.: Anfrage an http://www.google.com
  - Anfrage an einen Haupt-Nameserver (ns1 bis ns4.google.com)
  - Nameserver liefert Liste von *regionalen* IP-Adressen
  - Browser wählt eine davon für die HTTP-Anfrage
  - Hinter IP-Adresse verbirgt sich ein HTTP-Load-Balancer
  - Load-Balancer wählt einen Webserver aus regionalem Cluster
  - Anfrage wird von Webserver mittels weiterer Server (z.B. Index-, Dokument-Server) mit Hilfe diverser Techniken (z.B. Google-Filesystem 2, BigTable und MapReduce) bearbeitet
  - Am Ende erhält der User das Ergebnis

Zusammenfassung

Zusammenfassung

- Traffic-Engineering auf fast allen Netzschichten möglich
- Anwendungsmöglichkeiten breit gefächert
  - Verbesserung des QoS
  - Lösen von Routing-Problemen
  - Performance
  - Sicherheit
  - …
- Techniken – je nach Anwendungszweck – sehr vielfältig
  - QoS, Traffic-Shaping, Routing, NAT, Firewalling, Tunneling, Proxying, Load-Balancing, …
- ABER: Spannungsfeld Netzneutralität fast immer präsent!



Vielen Dank!