

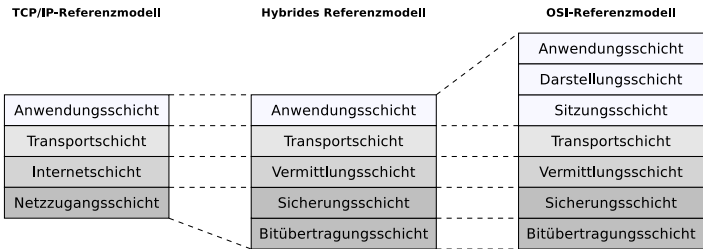
## 4. Foliensatz Computernetze

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences  
(1971–2014: Fachhochschule Frankfurt am Main)  
Fachbereich Informatik und Ingenieurwissenschaften  
[christianbaun@fb2.fra-uas.de](mailto:christianbaun@fb2.fra-uas.de)

# Sicherungsschicht

- Aufgaben der Sicherungsschicht (Data Link Layer):
  - Sender: Pakete der Vermittlungsschicht in Rahmen (Frames) verpacken
  - Empfänger: Rahmen im Bitstrom der Bitübertragungsschicht erkennen
  - Korrekte Übertragung der Rahmen innerhalb eines physischen Netzes gewährleisten durch Fehlererkennung mit Prüfsummen
  - Physische Adressen (MAC-Adressen) bereitstellen
  - Zugriff auf das Übertragungsmedium regeln



Übungsblatt 3 wiederholt die für die Lernziele relevanten Inhalte dieses Foliensatzes

- Geräte: Bridge, Layer-2-Switch (Multiport-Bridge), Modem
- Protokolle: Ethernet, Token Ring, WLAN, Bluetooth, PPP

# Lernziele dieses Foliensatzes

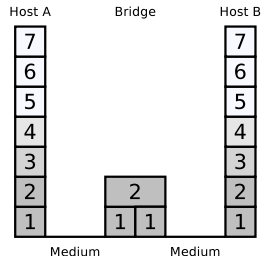
- Sicherungsschicht (Teil 1)
  - Geräte der Sicherungsschicht
    - Lernende Bridges
    - Kreise auf der Sicherungsschicht
    - Spanning Tree Protocol
    - Auswirkungen auf die Kollisionsdomäne
  - Adressierung in der Sicherungsschicht
    - Format der MAC-Adressen
    - Eindeutigkeit von MAC-Adressen
    - Sicherheit von MAC-Adressen

# Geräte der Sicherungsschicht: Bridges

- Geräte der Bitübertragungsschicht verlängern physische Netze
  - Sollen aber Rahmen von einem physischen Netz in andere weitergeleitet werden, sind **Bridges** nötig
- Eine Bridge hat nur 2 Schnittstellen
  - Solche Bridges verbinden meist Netzwerke, die auf unterschiedlichen Technologien (Übertragungsmedien) basieren  $\implies$  siehe Folien 6 und 7
- Einfaches Bridges leiten alle eintreffenden Rahmen weiter



- Bridges mit  $> 2$  Schnittstellen heißen **Multiport-Bridge** oder **Layer-2-Switch**
  - Sie haben typischerweise zwischen 4 und 48 Schnittstellen



# Arbeitsweise von Bridges und Layer-2-Switches

- Bridges und Switche untersuchen die Rahmen mit **Prüfsummen** auf Korrektheit
- Zum Filtern und Weiterleiten der Rahmen brauchen sie **keine Adresse**, da sie selbst nicht aktiv an der Kommunikation teilnehmen
  - Sie arbeiten wie die Geräte der Bitübertragungsschicht transparent
    - Grund: Sie kommunizieren nicht auf einer höheren Protokollschicht als der Sicherungsschicht

## Beispiel für Bridges im Alltag (1/2) – WLAN-Bridge



- Ermöglicht die Integration von Geräten mit RJ45-Netzwerkanschluss (z.B. Netzwerkdrucker, Desktops, Spielkonsolen, . . . ) in ein lokales Funknetz (WLAN)
- Verbindet ein kabelgebundenes Netzwerk mit einem Funknetz

## Beispiel für Bridges im Alltag (2/2) – Laser-Bridge

- Verbinden zwei Gebäude via Laserstrahl
  - Auf jedem Gebäude steht eine Laser-Sende-/Empfangseinheit
  - Interessante Alternative zu Kabeln (wenn Sichtkontakt besteht)

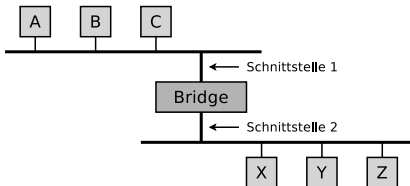


Bildquelle: <http://www.made-in-zelenograd.com> und <http://www.laseritc.ru>

Interessante Bauanleitung für eine eigene Laser-Bridge

- <https://hackaday.com/2017/04/19/go-wireless-with-this-diy-laser-ethernet-link/>
- <http://blog.svenbrauch.de/2017/02/19/homemade-10-mbits-laser-optical-ethernet-transceiver/>

# Lernende Bridges (1/2)



- Optimierung: **Lernende Bridges**
- Die Abbildung zeigt, dass es nicht sinnvoll ist, wenn eine Bridge alle Rahmen weiterleitet

- Kommt zum Beispiel ein Rahmen von Teilnehmer B für Teilnehmer A an Schnittstelle 1 der Bridge an, ist es nicht nötig, dass die Bridge diesen Rahmen über Schnittstelle 2 weiterleitet

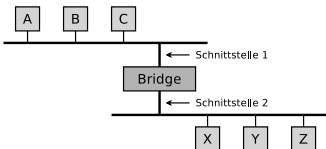
Gerät	Schnittstelle
A	1
B	1
C	1
X	2
Y	2
Z	2

- Bridges müssen lernen, welche Netzwerkgeräte über welchen Schnittstelle erreichbar sind
- Administratoren könnten die Tabellen in den Bridges pflegen
  - Das wäre sehr aufwändig
- Manuelle Eingriffe sind nicht nötig, da die Bridges ihre **Weiterleitungstabellen** selbst pflegen



# Lernende Bridges (2/2)

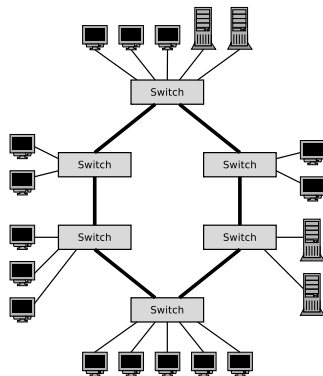
- Vorgehensweise:
  - **Bridges speichern die Absenderadressen der Rahmen, die sie erreichen**
    - Wenn Gerät A ein Rahmen an einen anderen Host sendet, merkt sich die Bridge, dass der Rahmen von Host A an Schnittstelle 1 einging
  - So füllt sich die Weiterleitungstabelle mit der Zeit mit Einträge, welche Netzwerkgeräte sich in den verbundenen physischen Netzen befinden



- Beim Hochfahren einer Bridge ist ihre Weiterleitungstabelle leer
  - Einträge werden im Laufe der Zeit erfasst
  - Jeder Eintrag hat ein **Verfallsdatum** (Time to Live – TTL)
    - Sie sind nur eine bestimmte Zeit gültig
- Die Weiterleitungstabelle ist nicht unbedingt vollständig
  - Das ist aber kein Problem, da sie zur **Optimierung** dient
    - Existiert für ein Netzwerkgerät kein Eintrag in der Weiterleitungstabelle, leitet die Bridge den Rahmen in jedem Fall weiter

# Kreise auf der Sicherungsschicht

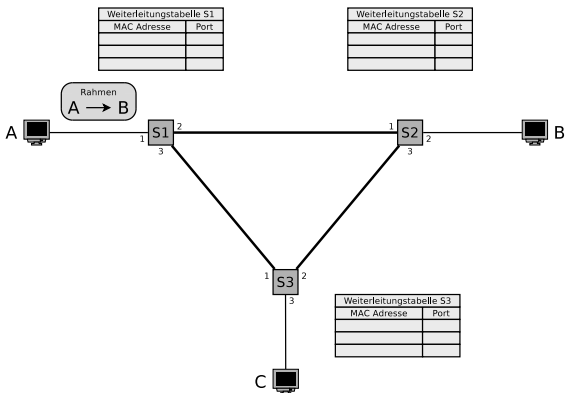
- Ein potentielles Problem sind **Kreise**
  - Computernetze sollten auf der Sicherungsschicht zu jedem möglichen Ziel immer nur einen Pfad haben
    - Das soll vermeiden, dass Rahmen dupliziert werden und mehrfach am Ziel eintreffen
  - Kreise können die Leistung des Netzes vermindern oder sogar zum Totalausfall führen
    - Andererseits dienen redundante Netzpfade als Backup für den Ausfall einer Leitung



## Gründe für das Entstehen von Kreisen auf der Sicherungsschicht

- Unachtsame Administratoren
- Absicht zum Ausgleich gestörter Verbindungen (Redundante Leitung)

# Beispiel für Kreise in einem LAN (1/6)

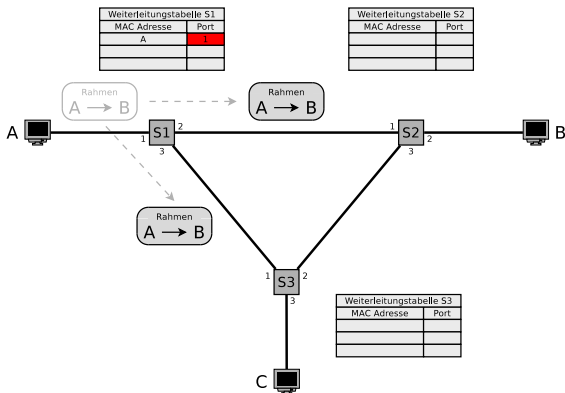


- Ein lokales Netz (LAN) hat Kreise auf der Sicherungsschicht
- Die Weiterleitungstabellen der Switche sind leer
- Im Beispiel will Knoten A einen Rahmen an Knoten B senden

Quellen für ähnliche Beispiele:

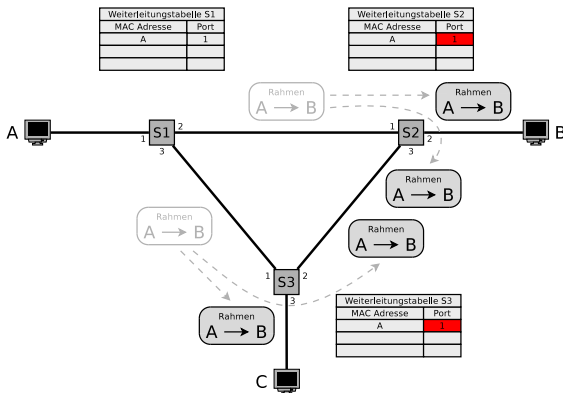
- Olivier Bonaventure. <http://cnp3book.info.ucl.ac.be/2nd/html/protocols/lan.html>
- Rüdiger Schreiner. Computernetzwerke. Hanser (2009)

# Beispiel für Kreise in einem LAN (2/6)



- Der Rahmen passiert Switch 1
- Switch 1 trägt den Port zu Knoten A in seine Tabelle ein
- Switch 1 kennt den Port zu Knoten B nicht
  - Darum sendet er Kopien des Rahmens über alle Ports (außer Port 1)

# Beispiel für Kreise in einem LAN (3/6)

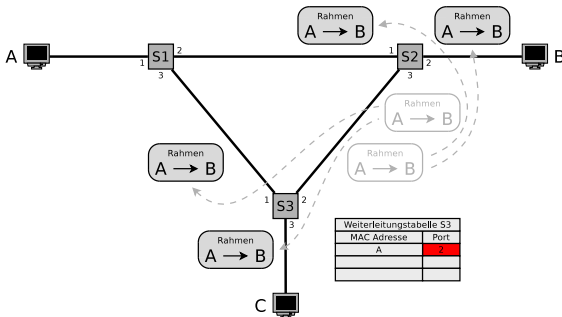


- Der Rahmen passiert Switch 2 und 3
- Switch 2 und 3 tragen den Port zu Knoten A in ihre Tabellen ein
- Switch 2 und 3 kennen den Port zu Knoten B nicht
  - Darum leiten Switch 2 und 3 Kopien des Rahmens über alle Ports weiter, außer über die Ports, an denen der Rahmen Switch 2 und 3 erreicht hat

# Beispiel für Kreise in einem LAN (4/6)

Weiterleitungstabelle S1	
MAC Adresse	Port
A	1

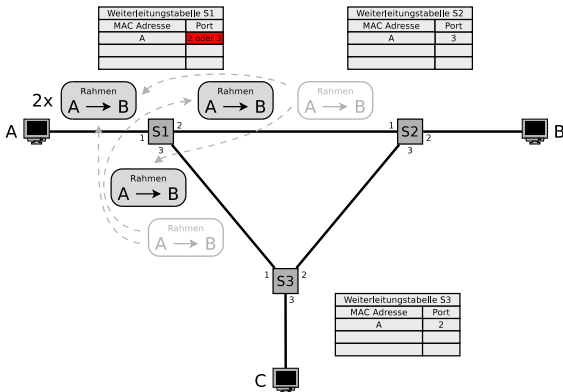
Weiterleitungstabelle S2	
MAC Adresse	Port
A	3



Weiterleitungstabelle S3	
MAC Adresse	Port
A	2

- Kopien des Rahmens passieren erneut Switch 2 und 3
- Switch 2 und 3 aktualisieren ihre Tabellen

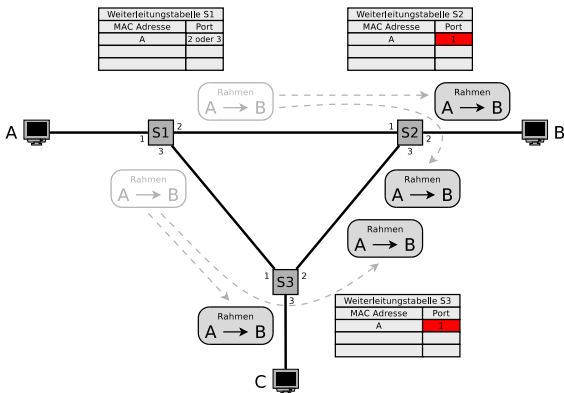
# Beispiel für Kreise in einem LAN (5/6)



- 2 Kopien des Rahmens erreichen Switch 1  
 ⇒ **Schleife!**
- Switch 1 sendet Kopien des Rahmens, die er über...
  - Port 2 empfangen hat an Port 1 und 3
  - Port 3 empfangen hat an Port 1 und 2
- Switch 1 aktualisiert seine Tabelle

- Die Reihenfolge, in der die Rahmen Switch 1 erreichen, kann nicht vorhergesagt werden

# Beispiel für Kreise in einem LAN (6/6)



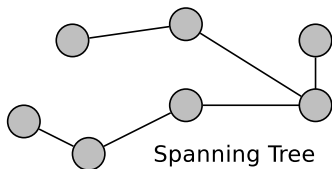
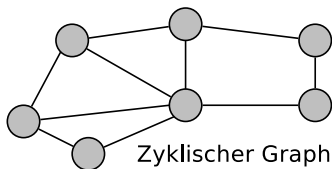
- Jeder Rahmen von Knoten A verursacht 2 Kopien, die endlos im Netz kreisen
  - Das Senden weiterer Rahmen durch Knoten A flutet das Netz und lässt es irgendwann zusammenbrechen

- Kopien des Rahmens passieren erneut Switch 2 und 3
- Switch 2 und 3 aktualisieren ihre Tabellen
- **Ethernet definiert keine TTL oder ein HopLimit**
  - Darum besteht die Schleife so lange, bis die Tabellen der Switche einen Eintrag für Knoten B enthalten



# Kreise im LAN handhaben

- Bridges müssen in der Lage sein, Kreise zu handhaben
- Lösung: **Spanning Tree Algorithmus**



- Ein Computernetz, das aus mehreren physischen Netzen besteht, ist ein Graph, der möglicherweise Kreise enthält
  - Der Spannbaum (Spanning Tree) ist ein Teilgraph des Graphen, der alle Knoten abdeckt, aber kreisfrei ist, weil Kanten entfernt wurden
  - Die Implementierung des Algorithmus ist das **Spanning Tree Protocol (STP)**

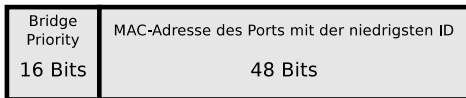


# Spanning Tree Protocol – Vorbedingung (1/2)

- Damit das STP funktionieren kann, muss jede Bridge eine eindeutige Kennung haben
  - Die Kennung (**Bridge-ID**) ist 8 Bytes lang
  - Es existieren 2 unterschiedliche Darstellungen der Bridge-ID

## 1 Aufbau der Bridge-ID gemäß IEEE

- Die Bridge-ID enthält die Bridge Priority (2 Bytes) und die MAC-Adresse (6 Bytes) der Bridge-Schnittstelle mit der niedrigsten Port-ID
  - Die Bridge Priority kann der Administrator selbst festlegen und hat einen beliebigen Wert zwischen 0 und 65.535
  - Standard-Wert: 32.768



# Spanning Tree Protocol – Vorbedingung (2/2)

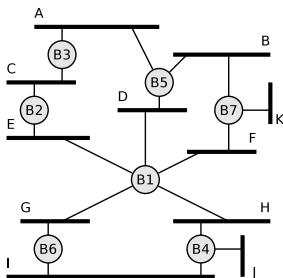
## ② Cisco-Erweiterung der Bridge-ID um die Extended System-ID

- Cisco ermöglicht mit seinen Bridges, dass jedes virtuelle LAN (VLAN) seinen eigenen Spannbaum aufbaut
- Dafür wird die ursprünglich 2 Byte große Bridge Priority unterteilt
  - 4 Bits kodieren nun die Bridge Priority
    - ⇒ Damit lassen sich nur 16 Werte darstellen
    - ⇒ Der Wert muss darum Null oder ein Vielfaches von 4.096 sein
    - ⇒ 0000 = 0, 0001 = 4.096 ... 1110 = 57.344, 1111 = 61.440
  - 12 Bits heißen **Extended System ID** und kodieren die VLAN-ID
    - ⇒ Der Inhalt stimmt mit dem VLAN-Tag im Ethernet-Rahmen überein
    - ⇒ Mit 12 Bits können 4.096 unterschiedliche VLANs adressiert werden



# Spanning Tree Protocol – Arbeitsweise (1/2)

- Die Nachrichten, mit denen die Bridges kommunizieren, heißen **Bridge Protocol Data Unit (BPDU)**
  - Sie werden im Datenfeld von Ethernet-Rahmen via Broadcast an die benachbarten Bridges gesendet

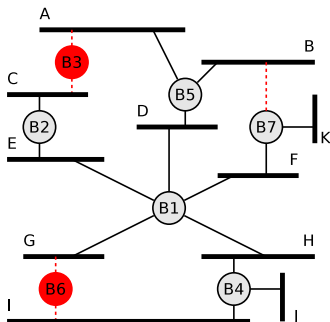


- Zuerst wählen die Bridges untereinander die Bridge mit der niedrigsten Bridge Priority in der Bridge-ID
  - Diese Bridge wird die **Wurzel** des aufzuspannenden Baums
  - Ist die Bridge Priority bei mehreren Bridges identisch, wird die Bridge mit der niedrigsten MAC-Adresse die Wurzel

# Spanning Tree Protocol – Arbeitsweise (2/2)

- Für jedes physische Netz wird eine direkt verbundene Bridge ausgewählt, die für die Weiterleitung der Rahmen in Richtung Wurzel zuständig ist
  - Diese Bridge ist die **designierte Bridge** für das betreffende Netz
  - Es wird immer diejenige Bridge ausgewählt, über die zu den **geringsten Pfadkosten die Wurzel erreicht** werden kann
    - Die Pfadkosten zur Wurzel sind die Summe der Pfadkosten der einzelnen physischen Netze auf dem Weg zur Wurzel

Datendurchsatzrate	Pfadkosten
10.000 MBit/s	2
1.000 MBit/s	4
100 MBit/s	19
16 MBit/s	62
10 MBit/s	100
4 MBit/s	250

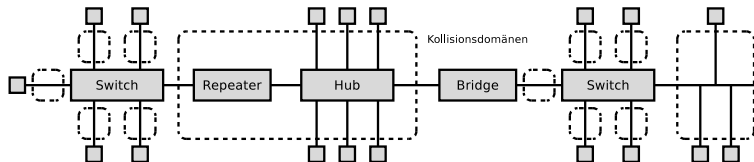


Die Pfadkosten sind durch die IEEE genormt, können aber manuell angepasst werden

Der Austausch der BPDU-Nachrichten wird in der Vorlesung nicht im Detail besprochen

## Kollisionsdomäne – Bridges und Layer-2-Switches

- Bridges und Switche arbeiten auf der Sicherungsschicht und leiten Rahmen von einem physischen Netz zu anderen
- **Jedes physische Netz ist eine eigene Kollisionsdomäne**
  - Unterteilt man ein physisches Netz durch eine Bridge oder einen Switch, unterteilt man auch die Kollisionsdomäne
    - Dadurch sinkt die Anzahl der Kollisionen
- Bei Bridges und Switches bildet jeder Port eine eigene Kollisionsdomäne



- In einem „vollständig geschwichten Netz“ ist mit jedem Port eines Switches nur ein Netzwerkgerät verbunden
  - Ein solches Netzwerk ist frei von Kollisionen und Stand der Technik

## Adressierung auf der Sicherungsschicht

- Die Protokolle der Sicherungsschicht definieren das Format der physischen Adressen
- **Endgeräte (Hosts), Router und Layer-3-Switche** benötigen zwingend physische Adressen
  - Diese Geräte müssen auf der Sicherungsschicht adressierbar sein, um Dienste auf höheren Schichten anzubieten
- **Bridges und Layer-2-Switche** nehmen nicht aktiv an der Kommunikation teil
  - Darum brauchen sie für ihre Basisfunktionalität, also das Filtern und Weiterleiten der Rahmen, keine physischen Adressen
  - Bridges und Switche benötigen dann physische Adressen, wenn sie das STP zur Vermeidung von Kreisen anwenden, oder Dienste aus einer höheren Schicht anbieten
    - z.B. Monitoring-Dienste zur Überwachung oder grafische Weboberflächen zur Administration
- **Repeater und Hubs**, die nur auf der Bitübertragungsschicht arbeiten, haben keine Adressen



## MAC-Adressen (1/2)

- **Physische Adressen** heißen **MAC-Adressen** (Media Access Control)
  - Sie sind unabhängig von den logischen Adressen der Vermittlungsschicht
- Ethernet verwendet das **Address Resolution Protocol** (ARP) um die logischen Adressen der Vermittlungsschicht (IPv4-Adressen) in MAC-Adressen aufzulösen
  - Bei IPv6 wird das **Neighbor Discovery Protocol** (NDP) verwendet, dessen Funktionalität identisch ist und das ähnlich arbeitet
- MAC-Adressen sind 48 Bits (6 Bytes) lang
  - Damit sind insgesamt  $2^{48}$  Adressen möglich
- Für eine kompakte und gut lesbare Darstellung sind MAC-Adressen meist in hexadezimaler Schreibweise geschrieben
  - Zudem sind die einzelnen Bytes durch Bindestriche oder Doppelpunkte voneinander getrennt
- Beispiel für diese Schreibweise: 00-16-41-52-DF-D7

## MAC-Adressen (2/2)

- Jede MAC-Adresse soll dauerhaft einem Netzwerkgerät zugewiesen und eindeutig sein
  - Es ist aber auch meist möglich, MAC-Adressen softwaremäßig zu ändern
    - Allerdings gilt die Änderung nur bis zum nächsten Neustart des Rechners
- **MAC-Broadcast-Adresse**
  - Will ein Netzwerkgerät einen Rahmen an alle anderen Geräte im gleichen physischen Netz senden, fügt es im Rahmen in das Feld der Zieladresse die Broadcast-Adresse ein
  - Bei dieser MAC-Adresse haben alle 48 Bits den Wert 1
  - Hexadezimale Schreibweise: FF-FF-FF-FF-FF-FF
  - Rahmen, die im Zielfeld die Broadcast-Adresse tragen, werden von Bridges und Switches nicht in andere physische Netze übertragen

# Eindeutigkeit von MAC-Adressen

- Das Institute of Electrical and Electronics Engineers (IEEE) verwaltet die ersten 24 Bits des MAC-Adressraums
  - Diese 24 Bits langen Teiladressen sind die **Herstellerkennungen** und heißen **MA-L** (MAC Address Block Large) bzw. **OUI** (Organizationally Unique Identifier)
  - Die Herstellerkennungen sind in einer Datenbank des IEEE einsehbar <http://standards.ieee.org/develop/regauth/oui/public.html>
- Die übrigen 24 Bits legen die Hersteller selbst für jedes Netzwerkgerät fest
  - Das ermöglicht  $2^{24} = 16.777.216$  individuelle Geräteadressen

MAC-Adressen	Hersteller	MAC-Adressen	Hersteller	MAC-Adressen	Hersteller
00-20-AF-xx-xx-xx	3COM	00-03-93-xx-xx-xx	Apple	00-0C-6E-xx-xx-xx	Asus
00-00-0C-xx-xx-xx	Cisco	00-50-8B-xx-xx-xx	Compaq	08-00-2B-xx-xx-xx	DEC
00-01-E6-xx-xx-xx	Hewlett-Packard	00-02-55-xx-xx-xx	IBM	00-02-B3-xx-xx-xx	Intel
00-04-5A-xx-xx-xx	Linksys	00-09-5B-xx-xx-xx	Netgear	00-04-E2-xx-xx-xx	SMC

- Kleinere Adressbereiche sind auch verfügbar: **MA-S** (MAC Address Block Small) und **MA-M** (MAC Address Block Medium)

# Sicherheit von MAC-Adressen

- In WLANs wird häufig mit einem MAC-Filter die MAC-Adresse als Zugangsschutz zur Basisstation (Access Point) verwendet
  - Im Prinzip ist das sinnvoll, da die MAC-Adresse das eindeutige Identifikationsmerkmal eines Netzwerkgeräts ist
- Der Schutz von MAC-Filtern ist aber gering, da MAC-Adressen softwaremäßig verändert werden können
  - Dieses Vorgehen heißt **MAC-Spoofing**

## Mit MAC-Adressen unter Linux arbeiten

- Eigene MAC-Adresse(n) auslesen: `ip link` oder `ifconfig`
- MAC-Adresse(n) vom Nachbarn (meistens der Router) auslesen: `ip neigh`
- MAC-Adresse setzen: `ip link set dev <Interface> address <MAC-Adresse>`
- Alternativ: `ifconfig <Interface> promisc`  
und dann: `ifconfig <Interface> hw ether <MAC-Adresse>`