INSTITUT FÜR
INFORMATIK
DER LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

LS Prof. Kranzlmüller

Praktikum Rechnernetze

Kapitel 2: Signalisierung und Rahmenbildung

MNM
TEAM
Munich Network Management Team

2 Signalisierung und Rahmenbildung

Inhaltsverzeichnis

Dieser Abschnitt beschäftigt sich mit der Bitübertragungsschicht und der Sicherungsschicht, den Schichten 1 und 2 im ISO-OSI Referenzmodell. In anderen Modellen sind diese Schichten zu einer “Netzzugangsschicht” zusammengefasst.

2.1 Bitübertragungsschicht (OSI-Schicht 1)

Bei jeder Interaktion zwischen zwei Rechnern müssen die Daten früher oder später eine physische Distanz überbrücken. Die binären Daten eines Rechners werden für den physischen Transport aufbereitet und anschließend über ein Medium an einen anderen Rechner übertragen. Kann der andere Rechner aus dem Empfangenen die ursprünglichen Daten rekonstruieren, ist es gelungen binäre Daten von einem Rechner an einen anderen zu übertragen.

Protokolle der Bitübertragungsschicht beschäftigen sich mit genau diesem Problem. Sie legen fest welches **Medium** benutzt wird und wie binäre Daten (Bits) als **Signale** auf das physische Medium **moduliert** werden. Dazu gehören mehrere Aspekte, die sicherstellen, dass alle Endpunkte auf die selbe Art und Weise Daten übermitteln und interpretieren. Diese Aspekte lassen sich in vier Gruppen unterteilen:

Physikalische Aspekte umfassen Eigenschaften des Mediums und der verwendeten Signale.

Mechanische Eigenschaften spezifizieren u.A. die Bauform der Anschlüsse an das Medium.

Funktionale Spezifikationen definieren die Benutzung des Mediums, z.B. Pin-Belegung und Takt.

Prozedurale Beschreibungen enthalten Elementarereignisse und deren Bedeutung, z.B. den genauen Ablauf zur Übertragung einer SDU.

Heute handelt es sich bei dem Medium meist um Kupfer, Lichtwellenleiter oder "Luft". Die Signale werden in der Regel so gewählt, dass sie deutlich voneinander unterscheidbar sind, da Signale durch Störeinflüsse leicht verfälscht werden können. Eine SDU auf Schicht 1 muss nicht genau ein Bit sein. Ein Schicht 1 Protokoll kann auch die parallele Übertragung von mehreren Bits gleichzeitig spezifizieren.

Durch äußere Störeinflüsse können Amplitude, Frequenz und Phase eines Signals bei der Übertragung verändert werden. Komponenten der Schicht 1 verarbeiten Signale dahingehend, dass eingehende Signale als Signalzustände entsprechend der Spezifikation aufgefasst werden (Diskretisierung). Die Umwandlung in Bits ist ein weiterer Arbeitsschritt, der in der Regel nur auf Komponenten, die auch eine Schicht 2 implementieren, durchgeführt werden muss. Gängige Schicht 1 Komponenten sind **Repeater**, Multiport-Repeater (**Hubs**) und Wavelength Division Multiplexer (**WDMs**).

2.1.1 Repeater und Hubs

Ein Repeater verfügt über genau zwei Anschlüsse. Ein eingehendes Signal auf einem Anschluss wird verstärkt auf dem anderen Anschluss ausgegeben. Repeater können eingesetzt werden, um das Problem der Dämpfung zu überwinden und Signale über längere Distanzen zu übertragen, als es die Sendeleistung des ursprünglichen Senders erlaubt. Die logische Weiterentwicklung des Repeaters ist der Hub. Dieser verfügt über mehrere Anschlüsse und gibt ein eingehendes Signal an allen anderen Anschlüssen verstärkt wieder aus.

2.1.2 Wavelength Division Multiplexer

Implementieren optische Sendestationen das selbe Schicht 1 Protokoll, so verwenden sie meist die selben Wellenlängen zur Signalisierung. Treffen diese Signale in einem gemeinsamen Medium aufeinander, entstehen Überlagerungen (Kollisionen), so dass kein Empfänger die ursprünglichen Daten rekonstruieren kann. Bei einem Aufbau, in dem mehrere Sender Signale auf dem selben Medium versenden, so dass Kollisionen entstehen können, spricht man von einer Kollisionsdomäne.

WDMs bilden mehrere eingehende optische Signale auf unterschiedliche, disjunkte Bereiche des Farbspektrums (Kanäle) einer ausgehenden Leitung ab (Multiplex). Dadurch

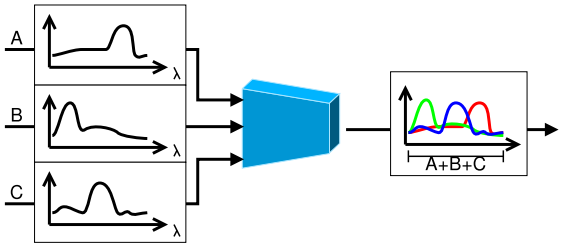

Abb. 2.1: Funktionsweise eines WDMs, der drei eingehende Signale auf ein gemeinsames Medium moduliert

können alle eingehenden Signale gleichzeitig über eine einzelne ausgehende Leitung übertragen werden, ohne Kollisionen zu erzeugen. Abbildung 2.1 zeigt einen WDM, der drei eingehende Signale A, B und C auf eine ausgehende Leitung moduliert. Diese Technik setzt man häufig ein, wenn der Aufwand zusätzliche Leitungen zu verlegen hoch ist. In Abbildung 2.1 sind A, B und C optische Signale der selben Schicht 1 Implementierung (z.B. 1 Gbps Ethernet, Monomode) und verwenden deshalb die selbe Wellenlänge für die Datenübertragung. Im WDM werden die Wellenlängen der eingehenden Signale modifiziert, so dass keine Überlagerung mehr stattfindet wenn die Signale in der ausgehenden Leitung aufeinander treffen.

Am anderen Ende der Leitung empfängt ein Demultiplexer das zusammengesetzte Signal. Dieser trennt das empfangene Signal auf, moduliert die Teilsignale zurück auf ihre ursprüngliche Form und sendet die Signale auf separaten Leitungen weiter.

2.2 Sicherungsschicht (OSI-Schicht 2)

Zu den Hauptaufgaben der Sicherungsschicht gehört Rahmenbildung (bzw. Blockbildung). Darunter versteht man die Gruppierung von Bits zu logischen Einheiten, den PDUs der Schicht 2 (Rahmen oder Blöcke, bzw. engl. frames oder blocks). Eine Basiskomponente der Schicht 2 ist die Bridge (bzw. “Brücke”). Eine Bridge ist eine Schicht 2 Komponente, die zwei Teilnetze (mit möglicherweise verschiedenen Übertragungstechniken) miteinander verbindet, also eine Brücke dazwischen bildet. Dazu muss sie eingehende Signale **interpretieren** und zu Rahmen zusammensetzen. Erst der vollständige Rahmen wird in das andere Teilnetz übertragen. Diese Technik heißt “store and forward”, da die Bridge eingehende Daten (Bits) speichert (store), bis der Rahmen vollständig empfangen wurde und erst im Anschluss daran den Rahmen weiterleitet (forward). Eine Bridge mit mehr als zwei Anschlüssen heißt Switch oder Multiport-Bridge.

2.2.1 Vergleich von Komponenten der Schichten 1 und 2

Aufgrund der Hauptaufgaben der Schichten 1 und 2 ergeben sich unterscheidende Merkmale der Komponenten dieser Schichten: Repeater, Hubs, WDM etc. auf der Schicht 1,

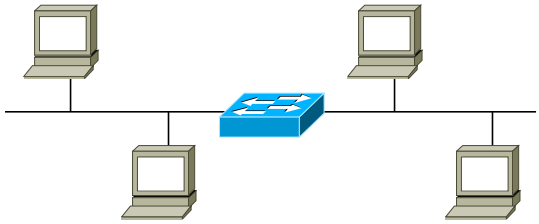

Abb. 2.2: Zwei Netze mit Bustopologie, verbunden über eine Bridge

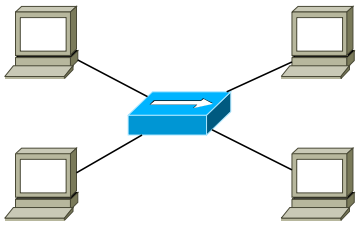

Abb. 2.3: Netz mit Sterntopologie

Bridges und Switches auf der Schicht 2.

Im Gegensatz zu einem WDM verarbeitet eine Bridge nicht einzelne eingehende Signale, sondern eingehende Rahmen. Die Bridge muss demnach in der Lage sein aus gespeicherten Informationen Rahmen zu bilden. Der Einsatz von store and forward ermöglicht es der Bridge unterschiedliche Bitübertragungstechniken für die beiden LANs zu verwenden, wogegen die Anschlüsse eines WDM den Festlegungen der physikalischen Signale entsprechen. Der Aufbau in Abbildung 2.2 erlaubt es unterschiedliche Schicht 1 Implementierungen “links” und “rechts” der Bridge zu nutzen.

2.2.2 Topologien

Neben der Rahmenbildung spezifizieren Schicht 2 Protokolle auch die Übertragung von Rahmen. Dazu gehören Vielfachzugriffsverfahren, die den Zugriff auf Schicht 1 bzw. auf gemeinsam genutzte Medien steuern, sowie die Übertragung von Rahmen zu Schicht 2 Endpunkten. Durch die vielfältigen Komponenten und Funktionen der Schichten 1 und 2 ergeben sich verschiedene Möglichkeiten einen physischen Aufbau eines Netzes (LAN) zu realisieren.

Abbildung 2.2 zeigt auf jeder Seite der Bridge ein Netz mit **Bustopologie**. Dabei sind mehrere Schicht 2 Endpunkte mit einem gemeinsam genutzten Medium verbunden. Ein Problem der Bustopologie ist die aufwändige Wartung. Tritt ein Hardwaredefekt auf, so kommt meistens das gesamte LAN zum Erliegen. Das Aufspüren der defekten Hardware ist schwierig, da jede Komponente für das Problem verantwortlich sein könnte. Das gemeinsam genutzte Kabel erstreckt sich meist über mehrere Räume oder auch Stockwerke und ist sehr aufwändig auszutauschen.

Eine andere Topologie, die bei diesem Problem hilft, ist die **Sterntopologie**. An die Stelle des gemeinsam genutzen Mediums tritt ein zentraler Hub (vgl. Abbildung 2.3). Da der Hub eingehende Signale auf alle angeschlossene Kabel repliziert, verhält sich ein Netz mit Sterntopologie genauso wie ein Netz mit Bustopologie; der Charakter des gemeinsam genutzten Mediums bleibt für die Signalübertragung erhalten. Deshalb kann an jeden Anschluss eines Hubs ein ganzes Teilnetz mit Bustopologie angeschlossen werden.

Der Vorteil des Hubs liegt darin, dass ein defektes Kabel an einem Anschluss nicht automatisch zu einem Zusammenbruch des gesamten Netzes führt. Außerdem kann an zentraler Stelle das fehlerhafte Kabel ermittelt werden. Üblicherweise ist an jeder Leitung eines Hubs ein einzelner Rechner angeschlossen, wodurch die Fehlerlokalisierung weiter vereinfacht wird. Der Defekt eines Kabels trennt so lediglich eine einzige Leitung (bzw. einen einzelnen Rechner) vom LAN, anstatt das gesamte LAN zum Erliegen zu bringen. Fällt der Hub aus, kann dieser leichter ausgetauscht werden, als ein Kabel, das durch mehrere Räume und Stockwerke verlegt wurde. Ersetzt man den Hub durch einen Switch, kann der Datenverkehr mit Bezug auf die Eigenschaften von Rahmen (Header) gesteuert werden. Ein Switch kann durch die Interpretation der eingehenden Informationen entscheiden eingehende Rahmen über wenige bestimmte Leitungen weitergeben, anstatt den Rahmen auf jedem Port zu replizieren.

2.3 Ethernet

Infolge der wachsenden Bedeutung der lokalen Vernetzung von Arbeitsplatzrechnern wurden zwischen 1972 und 1976 am *Xerox Palo Alto Research Center* die technologischen Grundlagen für ein gleichermaßen leistungsfähiges und “idiotensicheres” Local Area Network geschaffen. Dieses neue Local Area Network nannte man **Ethernet**, in Anspielung auf jenen geheimnisvollen “Lichtwellenäther”, welchen die Physiker des 19. Jahrhunderts so verzweifelt gesucht haben. Heute wird Ethernet formal als IEEE-Standard 802.3, CSMA/CD: Protokoll und physische Übertragungstechniken [IEEE 802.3], verwaltet und weiterentwickelt.

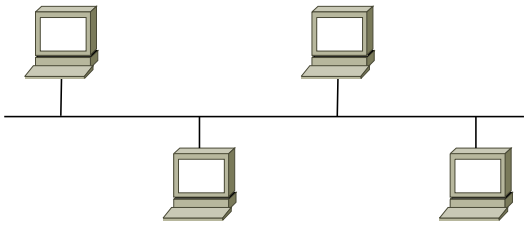

Abb. 2.4: Ethernet mit Bustopologie

Ethernet basiert in seiner ursprünglichen Form auf einer Broadcast-Technik, bei der alle Komponenten an das selbe Medium angeschlossen sind, wie in Abbildung 2.4 dargestellt. Sendet eine Komponente Daten, so werden diese von jeder anderen Komponente

empfangen. Senden mehrere Komponenten gleichzeitig, entsteht eine Datenkollision, so dass letztendlich keine Daten übertragen werden können. Bei einer Kollision treffen die Signale im Medium aufeinander, wodurch Interferenzen entstehen, so dass die ursprünglichen Signale nicht mehr unterscheidbar sind (siehe Kapitel 2.1). Aus diesem Grund benutzt Ethernet CSMA/CD, ein Verfahren um die Nutzung des gemeinsamen Mediums zu koordinieren. Die meisten heutigen Ethernet-Netze sind Sterntopologien, bei denen jeder Endpunkt (z.B. Rechner) exklusiv mit einem Switch-Port verbunden ist.

2.4 Virtuelle Topologien

Endeinrichtungen (Rechner) werden in der Praxis bestimmten Aufgaben oder Rollen einer Organisation zugeordnet, statt den Gegebenheiten der Vernetzung: Es ist z.B. wünschenswert, die Rechner nach Abteilungen bzw. Bereichen zu gruppieren. Oft entspricht dabei die physische Topologie, die durch die verlegten Medien (“Kabel”) gegeben ist, nicht den Nutzungsanforderungen eines LANs: z.B. werden Server in gemeinsam genutzten Server-Räumen untergebracht, in denen sich alle Server die selbe Leitung aus dem Raum hinaus teilen. Meist ist es jedoch so, dass sensible Daten nicht in andere LANs als dem abteilungseigenen LAN gelangen dürfen.

Hierzu können sogenannte *virtuelle LANs* (VLAN) benutzt werden, die eine logische LAN-Topologie auf eine physische aufbringen. VLANs können auf mehrere Arten erzeugt werden: durch die Gruppierung von Ports an einem Switch, durch Gruppierung von MAC-Adressen der zu einer Gruppe gehörenden Rechner und durch eine entsprechende Markierungen (*engl. tagging*) der gesendeten Rahmen. Im Praktikum liegt dabei der Schwerpunkt auf der zuletzt genannten Technik.

Ein wichtiger Standard in diesem Kontext ist der IEEE-Standard 802.1q [IEEE 802.1q], “Virtual LANs”. Dieser definiert das Anlegen, Betreiben und Verwalten von (mehreren) virtuellen LAN-Topologien innerhalb eines physischen LANs. Dazu wird jeder Rahmen einer virtuellen Infrastruktur mit einer für diese Infrastruktur eindeutigen Nummer (VLAN Identifier, VLAN-ID) in einem Feld (VLAN-Tag) im Ethernet-Header markiert. Netzkomponenten, die auf Schicht 2 operieren, können anhand der VLAN-ID Rahmen virtueller Topologien unterscheiden und unterschiedlich behandeln.

2.4.1 Monitoring-Port Konzept

Administrierbare Switches bieten meist die Funktion eines Monitoring-Ports. Dabei wird ein bestimmter Port des Switches ausgewählt, auf dem der Switch jeden Rahmen repliziert, der auf einem der anderen Switch-Ports empfangen wird. Diese Funktion ermöglicht es, jeden Rahmen, der vom Switch verarbeitet wird, an zentraler Stelle zu sammeln. Da die Übertragungsrate eines Monitoring-Ports meist deutlich kleiner ist als die aller anderen Ports zusammen, kann ein Monitoring-Port ab einer gewissen Auslastung nicht alle Rahmen replizieren. Aus diesem Grund werden Monitoring-Ports häufig nur zur Fehleranalyse eingesetzt. Das Konzept des Monitoring-Ports kann auch virtualisiert umgesetzt werden.

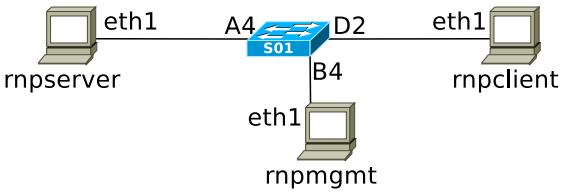

Abb. 2.5: Szenario für Management-Port Beispiel

Abbildung 2.5 zeigt ein Beispiel in dem die Rechner rnpserver und rnpclient über (rnpserver-eth1, S01-A-4) und (S01-D-2, rnpclient-eth1) verbunden sind. Außerdem ist der Rechner rnpmgmt mit dem Switch S01 verbunden. Wird der Switch-Port S01-B-4 als Management-Port konfiguriert, so empfängt der Rechner rnpmgmt an seiner Schnittstelle eth1 alle Rahmen, die rnpserver und rnpclient austauschen. Dies ermöglicht es die Interaktion dieser beiden Rechner zu überwachen, ohne direkten Zugang zu haben.

Das Einrichten eines Management-Ports ist eine spezielle Funktion, die von der Konfigurationssoftware der Switches bereitgestellt wird.

2.5 TUN/TAP Devices

Linux bietet seit Kernel-Version 2.2 die Möglichkeit virtueller Treiber-Schnittstellen an, sogenannte TUN/TAP Devices. TUN/TAP Devices existieren nur im Kernel und haben im Vergleich zu gewöhnlichen Schnittstellen keine physische Komponente. Falls das Betriebssystem Daten an ein TUN/TAP Device sendet, wird die Nachricht nicht an das physische Device, sondern an eine Benutzeranwendung, die über ein File-Descriptor mit dem TUN/TAP Device verbunden ist, weitergegeben. Was dann tatsächlich mit den Daten passiert, bleibt der Anwendung überlassen. Ein typisches Einsatzgebiet ist Tunneling, wie es beispielsweise in Virtual Private Networks (VPNs) der Fall ist. Durch den Einsatz von TUN Devices empfängt die VPN-Anwendung alle IP-Pakete, verschlüsselt deren Payload und umrahmt das ursprüngliche IP-Paket in ein weiteres IP-Paket mit aktualisierten Header-Informationen. Der Unterschied zwischen TUN und TAP Interfaces liegt darin, dass TAP Devices auf Schicht 2 und TUN Devices auf Schicht 3 operieren. Diese Unterscheidung ist wichtig, denn je nach Anwendungsfall ergeben sich entsprechende Anforderungen. Für VPN-Anwendungen bspw. sind TUN Devices auf Schicht 3 ausreichend. Eine ausführliche Dokumentation findet sich in der Linux-Kernel Dokumentation[1].

Im Rahmen dieses Praktikums setzen wir TUN/TAP Devices dazu ein, um unseren eigenen Netzwerk-Stack zu implementieren. Ein einfaches Beispiel findet sich in der beigelegten Mini-Anwendung[2]). Bevor ein TUN / TAP Device genutzt werden kann,

[1] `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/Documentation/networking/tuntap.txt`

[2] siehe `tuntap.c`

erzeugen wir zunächst ein virtuelles Device. Hierzu nutzen wir das allseits bekannte Tool iproute (`ip`).

```
$ ip tuntap help
   Usage: ip tuntap { add | del } [ dev PHYS_DEV ]
             [ mode { tun | tap } ] [ user USER ] [ group GROUP ]
             [ one_queue ] [ pi ] [ vnet_hdr ]

   Where: USER  := { STRING | NUMBER }
             GROUP := { STRING | NUMBER }
```

Nachdem ein physisches Device erzeugt wurde, setzen wir das Device zunächst auf den Status `up` und geben ihm anschließend ein Netz bzw. eine fixe Host-Adresse. Mit der richtigen Konfiguration wird jeglicher Datenverkehr durch das das TUN/TAP Device geleitet.

2.5.1 Virtuelle Schnittstellen unter Linux

Allgemein kann man sagen: Virtualisierung ist die Abstraktion von starren, beschränkenden Randbedingungen eines Systems zu konfigurierbaren Eigenschaften. Im Fall von virtuellen Schnittstellen bedeutet Virtualisierung, dass man die Anzahl der Schnittstellen eines Rechners verändern kann. Freilich lassen sich dadurch nicht mehr Kabel mit einem Rechner verbinden als physische Schnittstellen vorhanden sind. Trotzdem erhöhen sich die Möglichkeiten im Management.

Im Rahmen dieses Praktikums bieten virtuelle Schnittstellen den Mehrwert, dass man über eine physische Verbindung mehrere logische Verbindungen betreiben kann (vgl. Abbildung 2.6).

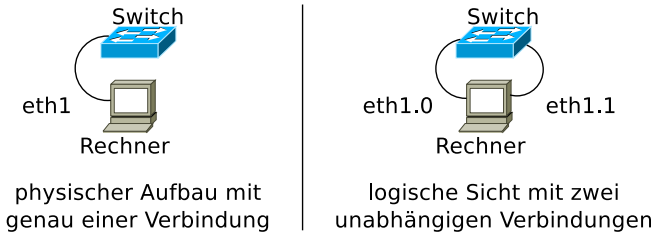

Abb. 2.6: Zwei logische Schnittstellen teilen sich eine Leitung

Jeder virtuellen Schnittstelle kann außer eigenen IP-Adressen auch eine eigene VLAN-ID zugewiesen werden. Es ist somit möglich, mit nur einer physischen Verbindung einen Rechner in mehrere virtuelle LANs einzubinden.

Es ist möglich einer einzelnen Schnittstelle mehrere IPs zuzuweisen und so einen Rechner in mehrere Subnets (z.B. 192.168.1/24 und 192.168.2/24) einzubinden. Der wesentliche Unterschied zum Einsatz virtueller Schnittstellen und VLANs liegt darin, dass virtuelle Schnittstellen bereits auf OSI-Schicht 2 implementiert werden, eine Trennung auf IP-Ebene jedoch eine Funktion von OSI-Schicht 3 ist.

Anlegen virtueller Schnittstellen mit VLAN-ID:
Das Anlegen von virtuellen Schnittstellen ist ebenfalls eine Funktion, die mit dem Befehl `ip` realisiert werden kann. Die Syntax zum Anlegen einer VLAN-Schnittstelle ist:

```
# ip link add link <pNIC> name <vNIC> type vlan id <VLAN-ID>
mit
   pNIC := Name der physischen Schnittstelle
   vNIC := Name der virtuellen Schnittstelle

z.B.:
# ip link add link eth0 name eth0.100 type vlan id 100
```

Zum Entfernen einer virtuellen Schnittstelle benutzen Sie:

```
# ip link del dev <vNIC>

z.B.:
# ip link del dev eth0.100
```

2.6 Scapy

Scapy ist ein Python Framework zur Inspektion und Manipulation von Paketen. Es erlaubt Ihnen Pakete vieler bereites implementierter Protokolle zu protokollieren, zu dekodieren, aus `pcap`-Dateien zu lesen, zu erstellen sowie diese zu versenden und vieles mehr. Scapy wurde auch für schnelles Prototyping entwickelt und benutzt Defaultwerte, die funktionieren.

Viele Aufgaben anderer Tools können von Scapy übernommen werden, z.B. Scanning, Traceroutes, Unit-Tests, Netzwerkerkennung und verschiedene Angriffe. Außerdem können Sie auch ungültige Rahmen versenden, eigene 802.11 Rahmen einbauen und verschiedene Techniken kombinieren (VLAN hopping+ARP cache poisoning, VoIP-Dekodierung auf einem WEP-verschlüsselten Kanal, etc.).

2.6.1 Installation auf der Infrastruktur

```
// On Debian
apt-get install python3-scapy

// On OpenWRT
opkg update; opkg install scapy
```

2.6.2 Usage

Eine kleine Übersicht:

- Verschiedene Protokolle und Header sind als Klassen implementiert, wie `IPv6` oder `ICMP`.
- Protokolle können ineinander geschachtelt werden mit dem Slash-Operator, z.B.: `IPv6()/TCP()` erstellt ein TCP-over-IPv6 Paket

- Pakete können mit den Methoden `show` und `show2` angezeigt werden, für das Senden bzw. Senden und Empfangen siehe `send()`, `sendp()`, `sr()`, `sr1()` und `srp()`.
- Die meisten Headeroptionen können verändert werden. Setzen Sie diese entweder im Konstruktor oder über Membervariablen, z.B.: `p = IP(ttl=64)`
- Nicht alle Optionen müssen angegeben werden, Scapy befüllt solche Werte mit Defaults
- Eine Liste aller Funktionen bekommen Sie über die Funktion `ls()`
- Für weitere Informationen und eine Demonstration, siehe `https://scapy.net/`

2.7 Aufgaben

Hinweis: vergessen Sie nicht Ihre Konfiguration in Netzplänen zu dokumentieren und auch die relevanten Ausgaben der verwendeten Programme zu übernehmen!

A200 Address Resolution and Neighbor Discovery (Theorie)

Das RFC 826 definiert das Address Resolution Protocol (ARP).
In RFC 4861 wird das Neighbor Discovery Protocol (NDP) spezifiziert.

i) Wozu wird ARP eingesetzt? Was ist der Unterschied zu NDP?

ii) Beschreiben Sie den Aufbau einer ARP-PDU und erläutern Sie die Bedeutung der einzelnen Felder!

iii) Welche unterschiedlichen ARP-PDUs gibt es? Welche NDP-PDUs gibt es?

iv) Wie lang (in Bytes) ist eine ARP-PDU in einem Netz in dem IPv4 und Ethernet eingesetzt werden?

v) Wie lang (in Bytes) ist eine Neighbor Solicitation Nachricht?

vi) Das RFC 826 spricht von einer Tabelle (table), deren Implementierung meist als ARP-Cache bezeichnet wird. Was soll laut RFC mit einer Ethernet-SDU passieren, wenn kein Eintrag zur Ziel-IP-Adresse in der Tabelle gefunden wird?

A201 VLANs nach IEEE 802.1q

Virtuelle Infrastrukturen dienen dazu Netze anzulegen und anpassen zu können, ohne physisch an den Geräten arbeiten zu müssen. Dies ist insbesondere in großen, weniger übersichtlichen Infrastrukturen von Vorteil.

Nachdem Sie in Aufgabe `A101` die Topologie der virtuellen Infrastruktur vollständig rekonstruiert haben, ändern Sie diese im Folgenden.

i) Modifizieren Sie die Topologie Ihrer virtuellen Infrastruktur so, dass die Router 1,2 und 3 ausschließlich Switches (Bridges) sind. Fügen Sie dieser Bridge alle Interfaces bis auf eth0 hinzu. (Hinweis: Benutzen Sie `ip link`, siehe Referenz[3])

   Die PCs 1–3 sowie Router 4 sind Hosts, die in *einem* (Sub-)Netz gemäß der Baum-Topologie in Abbildung 2.7 verbunden sind. Es genügt hierbei, wenn Sie überflüssige Links als `down` markieren.

ii) Testen Sie ihre Verbindungen zwischen den PCs 1–3 sowie Router 4 mittels `ping`. Router 1–3 haben per Definition von Bridges *keine* IP-Adresse. Vergeben Sie IPv4-Adressen aus Ihrem Adressraum.

[3] `https://wiki.archlinux.org/title/Network_bridge`

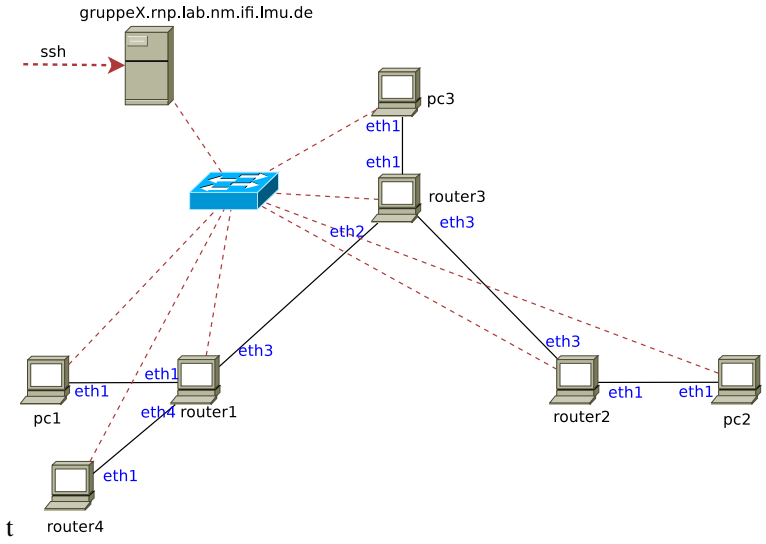

Abb. 2.7: Baum-Topologie

iii) Derzeit befinden sich alle Hosts im selben Netz. In dieser Aufgabe soll das Netz in zwei logisch getrennte VLANs (Schicht 2) aufgeteilt werden. Ziel ist es, dass Router 4 und pc3 in einem VLAN sowie pc2 und pc1 in einem anderen VLAN voneinander isoliert sind. Die *Schicht 3*-Adressen bleiben davon unberührt. Das Ergebnis soll sein, dass ausschließlich innerhalb des VLANs kommuniziert werden kann. Heißt also, dass bspw. Router 4 nur noch mit pc3 kommunizieren kann, allerdings nicht mehr mit den PCs 1–2 (obwohl diese noch im selben Subnetz auf Schicht 3 liegen).

   Testen Sie Ihre Konfiguration mittels `ping` und weisen Sie durch Mitlauschen via `tcpdump` (auf einem der Router 1–3), nach, dass bei den ICMP-Anfragen anhand der VLAN-IDs innerhalb des VLANs kommuniziert wird.

`A202` Analyse von ARP und NDP

Verwenden Sie die gleiche Topologie wie in der vorherigen Aufgabe. Entfernen Sie alle VLANS, sofern welche vorhanden sind.

i) Erläutern Sie die Ausgabe von `ip neigh` auf pc1!

ii) Leeren Sie den ARP-Cache von pc1 und pc2! (Hinweis: `ip neigh`)

iii) Stellen Sie sicher, dass die Interfaces der PCs Link-Local IPv6-Adressen haben. (Hinweis: `ip link [down/up]`)

iv) Benutzen Sie `ping` und `ping6` um IP-Verkehr zwischen pc1 und pc2 zu erzeugen. Wie verändert sich die Ausgabe von `ip neigh`?

v) Beobachten Sie mit `tcpdump` die Auflösung von IP- zu MAC-Adressen. An wel-

che MAC-Adressen werden die Anfragen zur Adressauflösung jeweils geschickt? Handelt es sich um Broadcasts?

vi) Fügen Sie Ihren PCs zusätzlich IPv6-Adressen aus folgenden Netz hinzu: `2001:db8:<Gruppennummer>::/64`

vii) Wiederholen Sie den IPv6-Versuch mit den neu hinzugefügten globalen Adressen. Erläutern Sie die Unterschiede!

viii) Vergleichen Sie die Ausgabe von `tcpdump` mit `scapy`. Lassen Sie sich die Details eines ARP/Neighbor Solicitation (NS) Pakets mit `scapy` ausgeben.

ix) Verwenden Sie `scapy` um sowohl ein ARP- als auch ein NS-Paket von pc1 zu versenden. (Hinweis: `Ether` und `sendp()`)

Erhalten pc2 und pc3 diese Pakete? Wie reagieren diese auf die Anfragen? Betrachten Sie die Antworten, die pc1 erhält!

x) Senden Sie jeweils ein ARP Reply und ein Neighbor Advertisement (NA) Paket mit einer neuen MAC-Adresse von pc1 zu pc2 mit `scapy`. Wie verhält sich der Neighbor cache auf pc2 nach deren Eingang?

A203 Implementierung von ARP in C

Die Aufgabe ist eine eigenständige Implementierung des ARP-Protokolls (vgl. RFC 826). Ziel ist es, dass Sie auf Basis von TUN/TAP Devices den Datenverkehr innerhalb eines Netzes abfangen und auf ARP-Anfragen mittels des Tools `arping` eine semantisch und syntaktisch korrekte ARP-Antwort zurück senden.

i) Machen Sie sich mit der Funktionsweise von TUN/TAP Devices vertraut und prüfen Sie mittels der Mini-Applikation `tuntap.c`, ob Ihre Konfiguration korrekt funktioniert.

ii) Überlegen Sie sich ein geeignetes Interface für ARP-Anfragen bzw. -Antworten und dokumentieren es in einem Header-File. Lesen Sie hierzu RFC 826[4].

iii) Implementieren Sie ihr spezifiziertes Interface so, dass eine `arping` Anfrage korrekt beantwortet wird. Sie können für Ihre Antwort eine beliebige IP-Adresse bzw. MAC-Adresse vergeben.

Beispiel für eine `arping` Anfrage:

```
$ arping -I mytap <ip>
```

◇

[4] `https://tools.ietf.org/html/rfc826`

Literatur

[IEEE 802.1q] *IEEE Std 802.1q-2005*, Mai 2006.

[IEEE 802] *IEEE Std 802-2001*, Februar 2007.

[IEEE 802.3] IEEE 802-3 WORK GROUP: *IEEE Std 802.3-2005*, Dezember 2005.