Kommunikationssysteme

Eine Aufgabensammlung für das Netzwerklabor[1]

Duale Hochschule Baden-Württemberg Stuttgart

Erika Herzer[2]
und
Armin Herzer[3]

März 2009

---

[1] Teile der Einführung in Linux stammen vom FSR Informatik der Uni Trier (http://cip.uni-trier.de/~fsrinfo/)

[2] erika@herzer-online.de
[3] armin@herzer-online.de

Linux ist

- seit 1992 als Download verfügbar (Version 0.11 in gepackter Form gerade man knapp 100 KB)
- "nur" ein Betriebsystem (-kern)
- mit Unix verwandt (Minix als Vorlage)
- Grundlage vieler Distributionen
  - z.B. Slackware, Debian, Ubuntu, Kubuntu, ...

▸ Distributionen sind

- Sammlungen von getesteten und aufeinander abgestimmten Anwendungen
- häufig kostenlos verfügbar (Download im Internet)
- manchmal auch käuflich zu erwerben

Microsoft gives you Windows. Unix gives you the whole house.
Einrichten muß man das Haus aber schon selbst :-)

- Stabilität der Daten und Rechnersysteme
- Benutzer und Prozesse geschützt
- Quelltext frei verfügbar (Open Source)
- "Einfache" Administration (wenn eingearbeitet)
- Kosten für Anschaffung und Erweiterungen niedrig
- Multiuser-, Multitasking-, Multisession-Betriebssytem
- schon immer Netzwerkfähig (v.a. Unix)
- Läuft auf nahezu allen gängigen CPUs
- Resourcenschonend (kann auch problemlos auf älterer Hardware betrieben werden)

There's no bug in Windows - that Linux can't fix!

Unter einem Live-System versteht man ein Betriebssystem, das ohne Installation gestartet werden kann. Das gesamte Betriebssystem wird hierzu auf einen bootfähigen Speicher wie USB-Stick, Flash-Speicher, Diskette, CD-ROM oder DVD kopiert.
Einige Einsatzbereiche von Live-Systemen:

- Programme testen
- bestehendes System reparieren (z.B. Viren entfernen)
- völlig sicherer Internetzugang (auf dem Rechner wird nichts mitgeloggt)

Live-Systeme können sehr leicht an individuelle Vorgaben angepasst werden!

Rang	Distribution	H.P.D*
1	Ubuntu	2305▼
2	openSUSE	1814▲
3	Mint	1351▼
4	Fedora	1351▼
5	PCLinuxOS	1088▬
6	Debian	1029▲
7	Mandriva	944▬
8	Dreamlinux	669▲
9	Damn Small	639▬
10	Sabayon	606▼
11	CentOS	566▬
12	Slackware	533▼
13	Kubuntu	483▼
14	Puppy	481▬
15	Gentoo	473▬
16	Arch	459▼
17	MEPIS	432▼
18	FreeBSD	415▼
19	Zenwalk	405▼
20	KNOPPIX	394▼
21	gOS	354▬
22	PC-BSD	338▲
23	sidux	325▲
24	Vector	313▬
25	Xubuntu	289▬
26	Elive	275▲
27	OpenSolaris	250▲
28	OpenGEU	244▬
29	Slax	243▼

http://www.distrowatch.com

Daten vom 24.09.08

233	SLAX
199	Kanotix
92	NimbleX
89	PCLinuxOS
52	Knoppix
46	GoblinX
40	Damn Small Linux
36	MEPIS
33	Puppy Linux
31	Ubuntu
15	FreeSBIE
15	Gentoo
10	dyne:bolic
10	Ultimate Boot CD
9	SLAX KillBill Edition
9	WHAX
7	Elive
7	GoblinX Mini Edition
6	FreeBSD LiveCD
6	Suse Live-Eval
6	SystemRescueCD
5	BeatrIX Linux
5	GamesKnoppix
5	INSERT
5	Knoppix STD
5	LLGP
4	BackTrack
4	GoboLinux
4	Kubuntu

http://www.livecdlist.com/

Im Prinzip kann aus jeder Linux-Distribution eine Live-Version erstellt werden. Was man dazu machen muß ist auf www.linux-live.org nachzulesen.

Slax bietet folgende Vorteile:

- große Benutzergemeinde und sehr gutes Forum
- wird ständig weiterentwickelt
- modularer Aufbau (d.h. zusätzliche Software wird als gepacktes Archiv (Modul) in ein Verzeichnis kopiert und kann sofort, auch on the fly, benutzt werden)
- hochoptimierte Kompressionsalgorithmen sparen Platz und sorgen für schnellen Zugriff auf den Inhalt der Module

Slax - your pocket operating system

Slax is a modern, portable, small and fast Linux operating system with a modular approach and outstanding design. Despite its small size, Slax provides a wide collection of pre-installed software for daily use, including a well organized graphical user interface and useful recovery tools for system administrators.

The modular approach gives you the ability to include any other software in Slax easily. If you're missing your favourite text editor, networking tool or game, simply download a module with the software and copy it to Slax, no need to install, no need to configure.

If you are a beginner or you're just too busy to make it yourself, follow a few steps to build your own customized operating system by using web-based interface here.

system tools and settings

Slabix wird seit dem Frühjahr 2007 auf der Basis einer aktuellen stabilen Slax-Version (z. Zt. 6.0.9) entwickelt[a]:

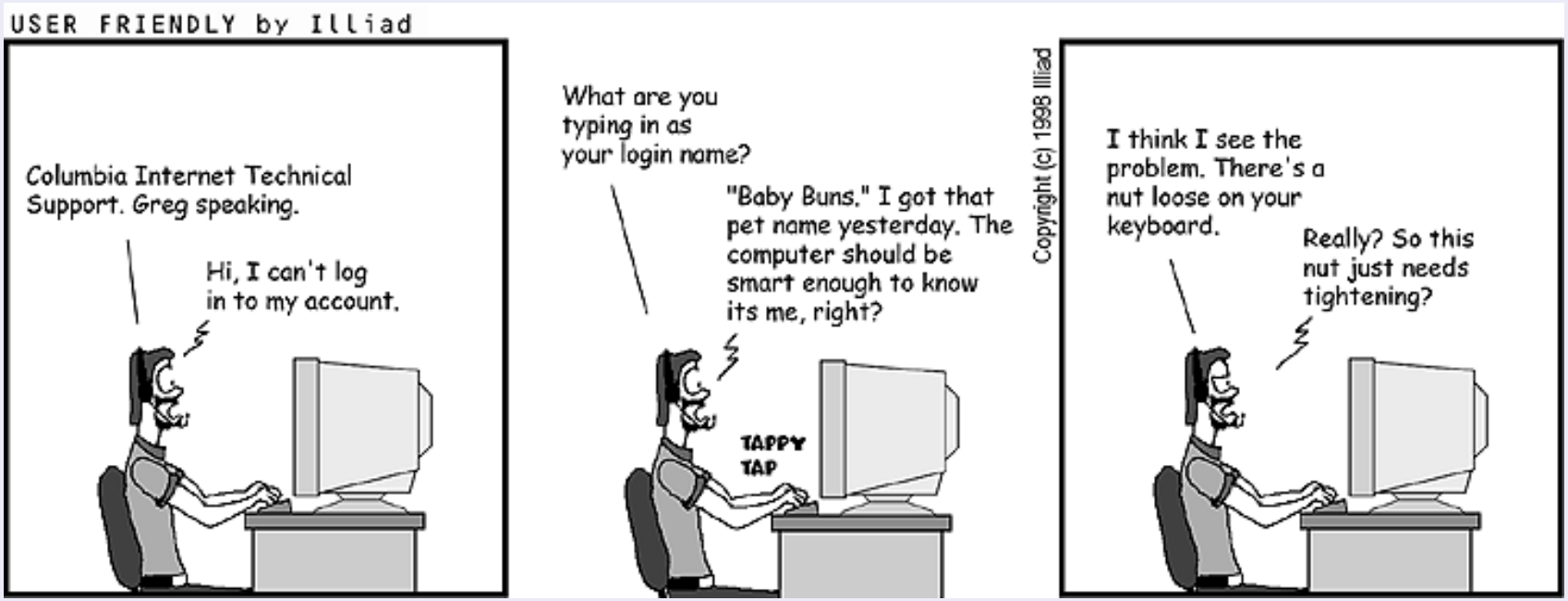
- Grundlage ist das ISO-Image von Slax
- alle Veränderungen finden ausschließlich im Ordner `modules` statt (der im Originalzustand leer ist)
- ca. 1.9 GB komprimierte Zusatzsoftware
  - selbst kompiliert und angepasst
  - von der Slax-Seite heruntergeladen
  - packages.slackware.it
  - www.slacky.eu
  - www.zenwalk.org
  - aus den Paket-Repositories anderer Distributionen (v.a. Ubuntu, Debian und Redhat)

[a] Stand: Winter 2008/09

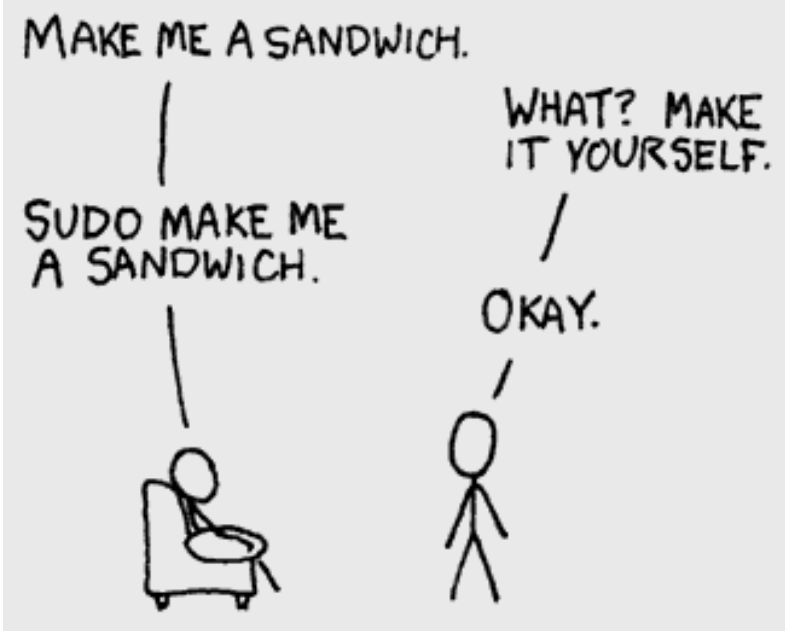
Durch den modularen Aufbau ist es sehr einfach die Distribution auf neuestem Stand zu halten und so auszustatten, wie es der Laborbetrieb erfordert:

- TeX Live 2008 (ca. 1GB komprimiert!!!)
- aktuellste Versionen von z.B. nmap und wireshark
- ein Sack voller Spiele :-)
  - Netzwerktechnik (Client/Server) z.B. pioneers, xskat oder ganz neu Blokus
  - HexTD als Java-Source zum selber übersetzen
  - diverse Schachengines (deren Quellcode offenliegt und die Spielstärken bis 2900 ELO-Punkten haben)
- KDevelop als Entwicklungsumgebung (wer's braucht)
- LabVIEW 6.1

- Unix-Systeme unterscheiden Benutzer
- Um den Rechner zu verwenden, muß sich der Benutzer beim System anmelden
  - Mittels Login/Passwort-Kombination werden die Benutzer identifiziert!
  - Login-Name und ein erstes Passwort werden vom Systemadministrator vergeben
  - Das Passwort ist ein privates Geheimnis und sollte unmittelbar nach dem ersten Login, geändert werden!



- Die Shell ist ein konfigurierbarer Kommandozeilen-Interpreter!
- Kann Kommandos ausführen und Programme starten.
- Stellt Eingabeaufforderung (Prompt) zur Verfügung, z.B.
  `herzer@herzer-desktop:~/dv1/u7$`
- Wichtigstes Programm in Unix-Systemen!
- Große Auswahl unter vielen Shells: bash (Standard), tcsh, zsh,

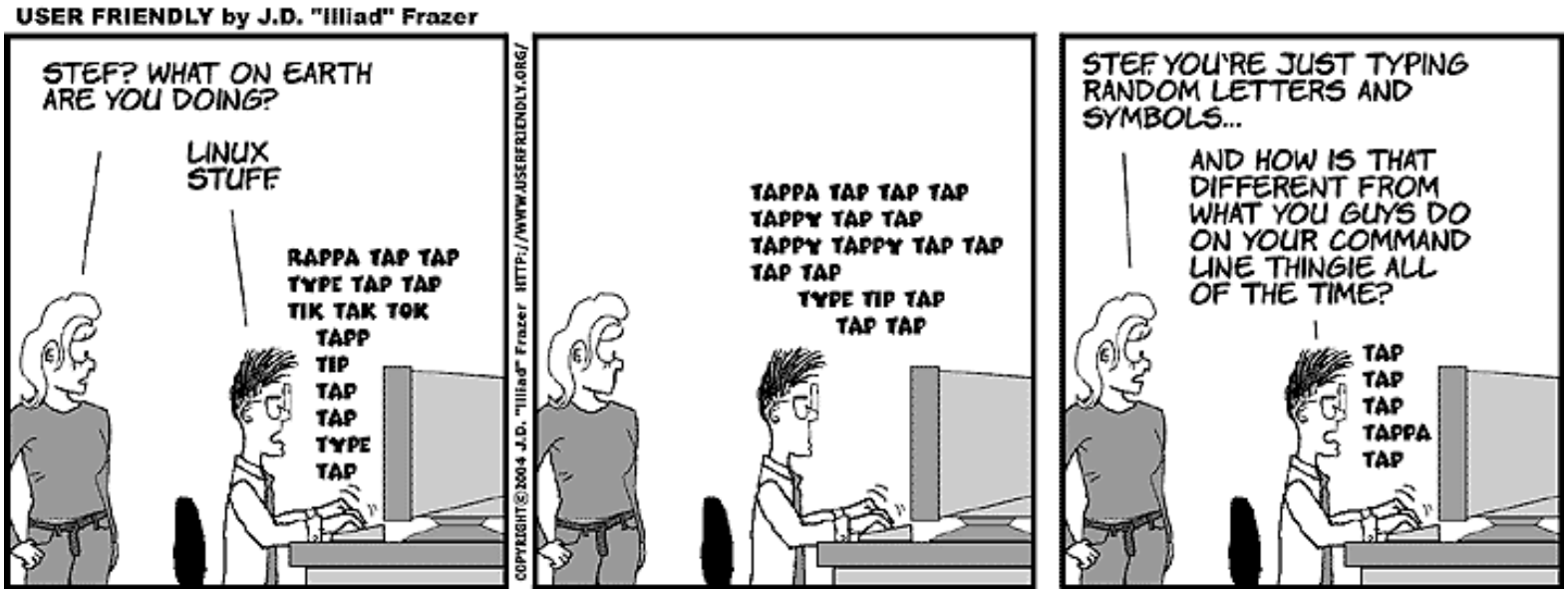


- Aktionen auslösen durch Eingabe entsprechender Kommandos, z.B.
  - *cd* - Change Directory
  - *ls* - LiSt directory (vgl. *dir*)
  - mv - MoVe file (Datei verschieben oder umbenennen)

- Allgemeine Form eines Kommandos:

  kommando␣[-option(en)]␣[argument(e)]↵

  ls␣-laF␣*.jpg↵

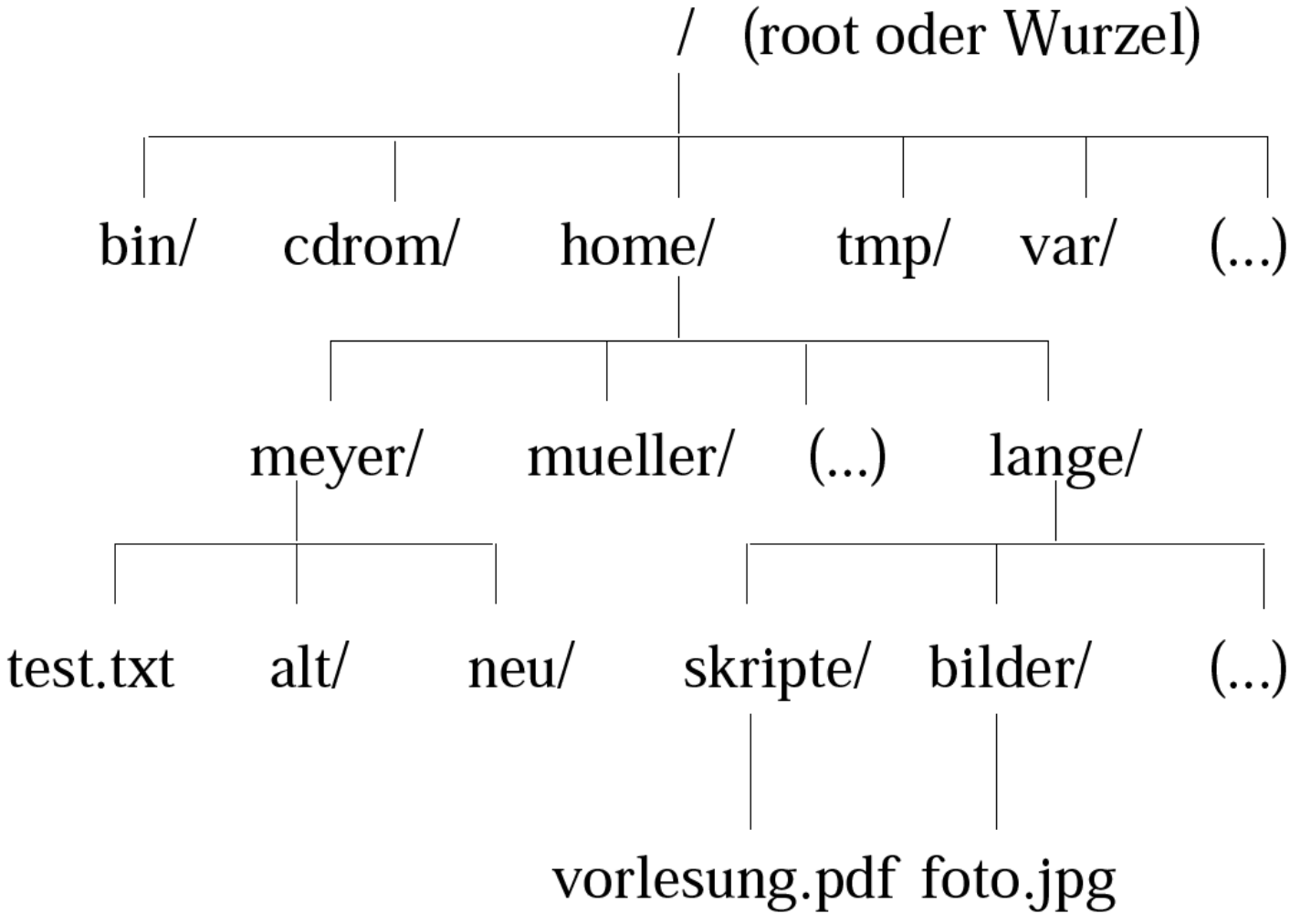


Wer nicht mehr weiter weiß, sollte unbedingt erst einmal die man-Pages lesen!

- Hilfe zu einem Kommando:

  man kommando bzw. *kommando –help*
- RTFM - Read The F****** Manual
- www.gnu.org
- ggf. im Quellcode-Paket nachsehen

- Das Dateisystem besteht aus hierarchisch angeordneten Dateien und Verzeichnissen
- Es gibt nur einen(!) Verzeichnisbaum mit einer Wurzel

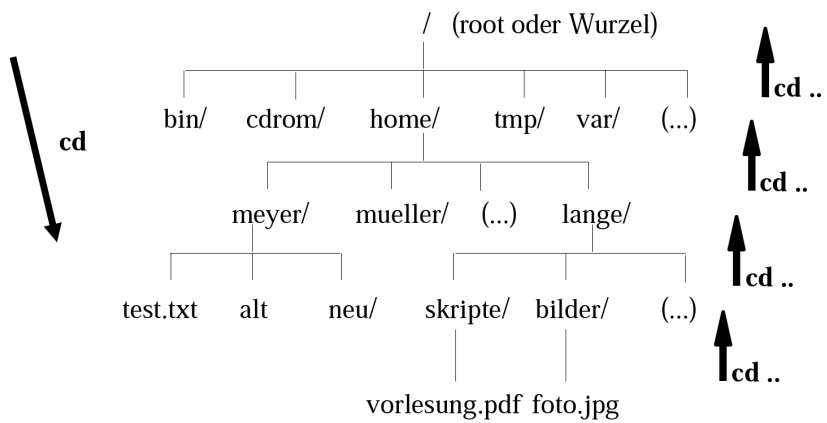


- Pfade im Dateisystem
  - Absolute Pfadangaben: Alle Verzeichnisnamen von der Verzeichniswurzel bis zum Ziel werden angegeben, z.B.

    ls /usr/ oder *cd /home/lange/bilder*
  - Relative Pfadangaben: Alle Verzeichnisnamen von der aktuellen Position im Dateisystem bis zum Ziel werden angegeben, z.B.

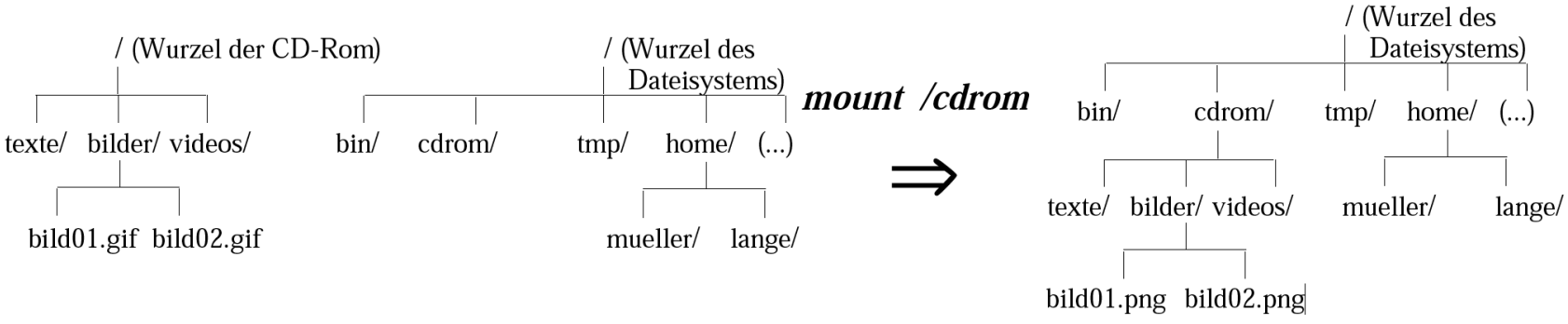
    ls bilder/ oder *cd ../meyer/*

- Spezielle Verzeichnisse
  - ~ bezeichnet des Home-Verzeichnis des aktuellen Benutzers
  - .. bezeichnet die Verzeichnisebene über dem aktuellen Verzeichnis (auch ../.. usw.)
  - . bezeichnet das aktuelle Verzeichnis



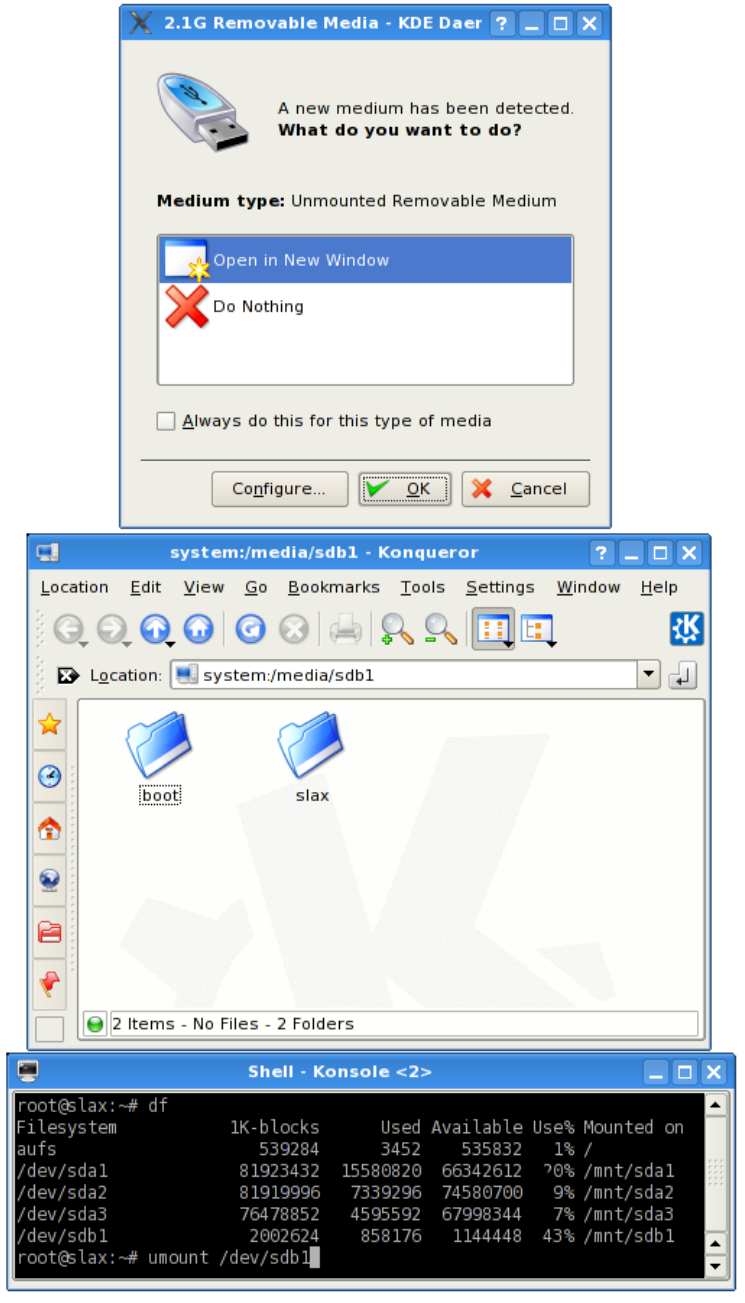
Unix kennt keine Laufwerksbuchstaben!

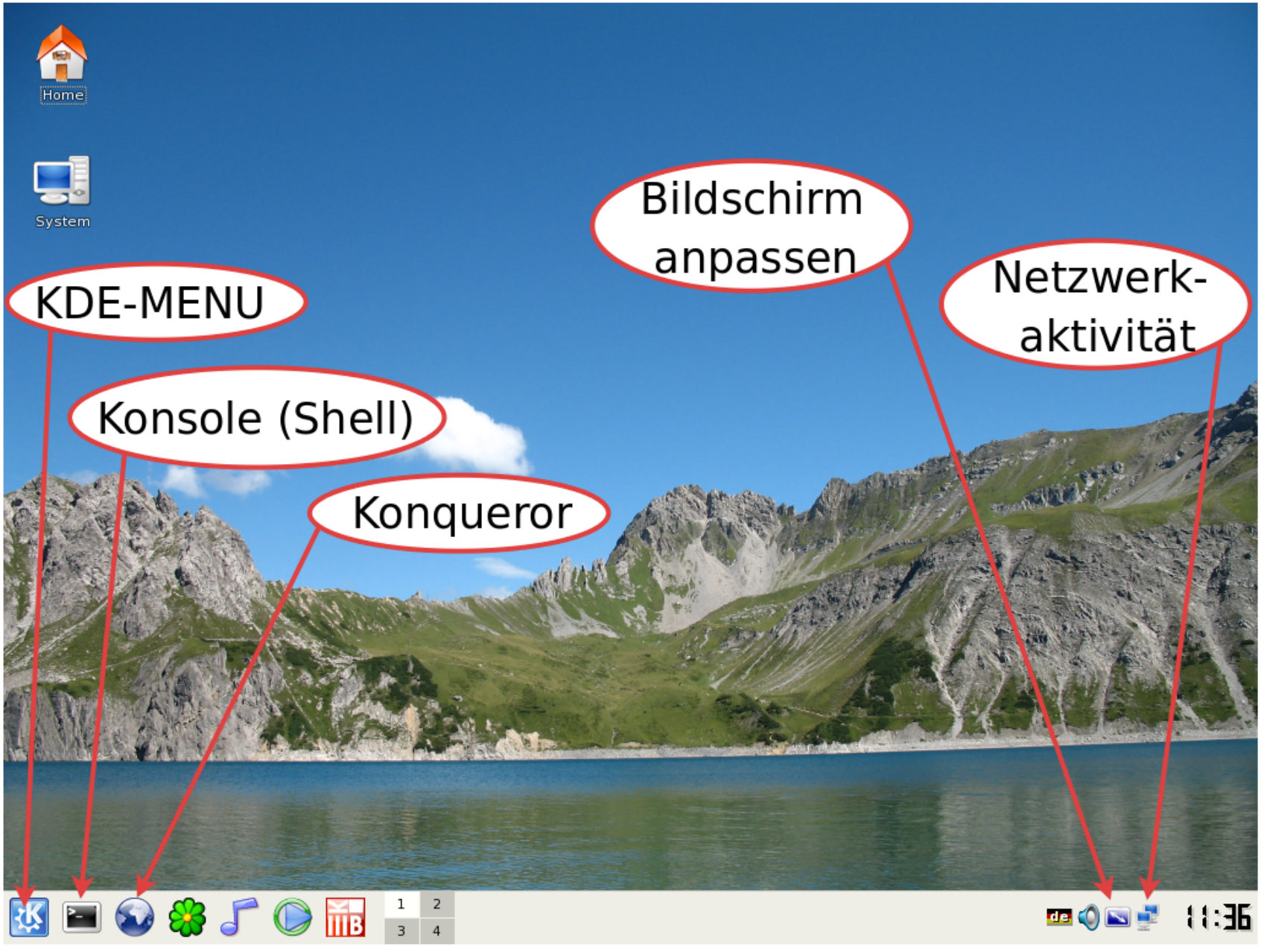
- Alle Laufwerke werden in den Verzeichnisbaum eingebunden
- Der Inhalt des Mediums ist dann in einem Unterverzeichnis (Mountpoint) verfügbar
- Einbinden: `mount` *<mountpoint>*
- Jedes Medium besitzt einen eigenen Mountpoint
- Entfernen: `umount` *<mountpoint>*. Erst danach kann/darf das Medium (DVD, USB-Stick, ...) entfernt werden!



USB-Sticks verwenden:

- **Mounten:** USB-Stick anschließen
- Dialogfenster mit OK bestätigen: Jetzt kann der Stick wie ein ganz normales Laufwerk benutzt werden
- **Unmounten:** Mit `df` den mountpoint in einer Konsole identifizieren (hier /dev/sdb1 bzw. /mnt/sdb1)
- umount /dev/sdb1 bzw. /umount/mnt
- `device is busy`: Eine Applikation greift noch auf den USB-Stick zu
- **WICHTIG:** USB-Stick immer unmouten (mit `df` überprüfen) und erst danach physikalisch entfernen!!!



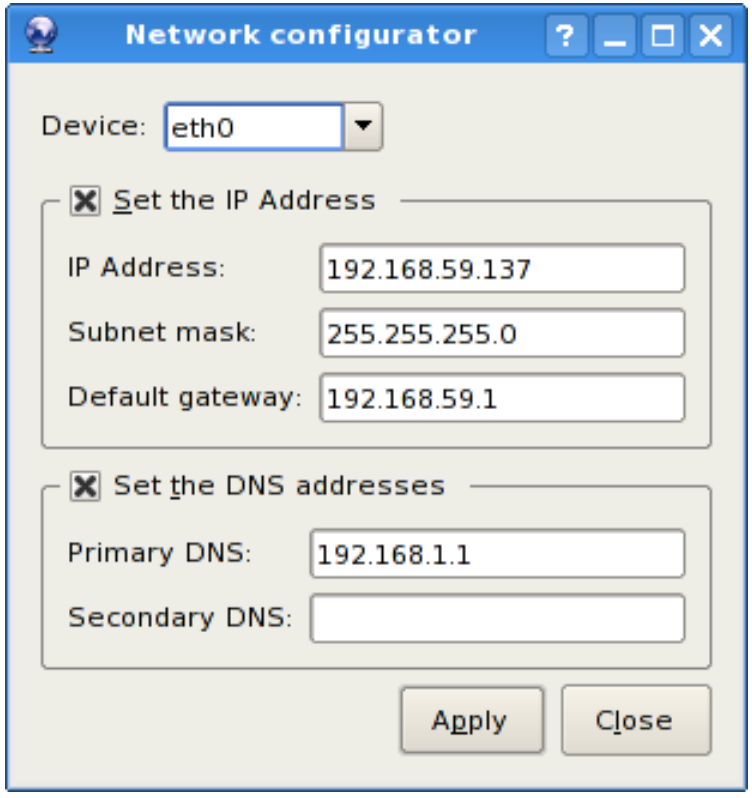

Home
System
KDE-MENU
Konsole (Shell)
Konqueror
Bildschirm anpassen
Netzwerk-aktivität
1
2
3
4
11:36

Über das `KDE-Menu` und den Menupunkt `Internet` wird das Programm `Set IP address (network-conf)` gestartet. Je nach Veranstaltung gibt es nun zwei Möglichkeiten:

- Entweder Konfigurationsparameter die PC am angebracht sind verwenden
- PC so konfigurieren wie es das Experiment erfordert

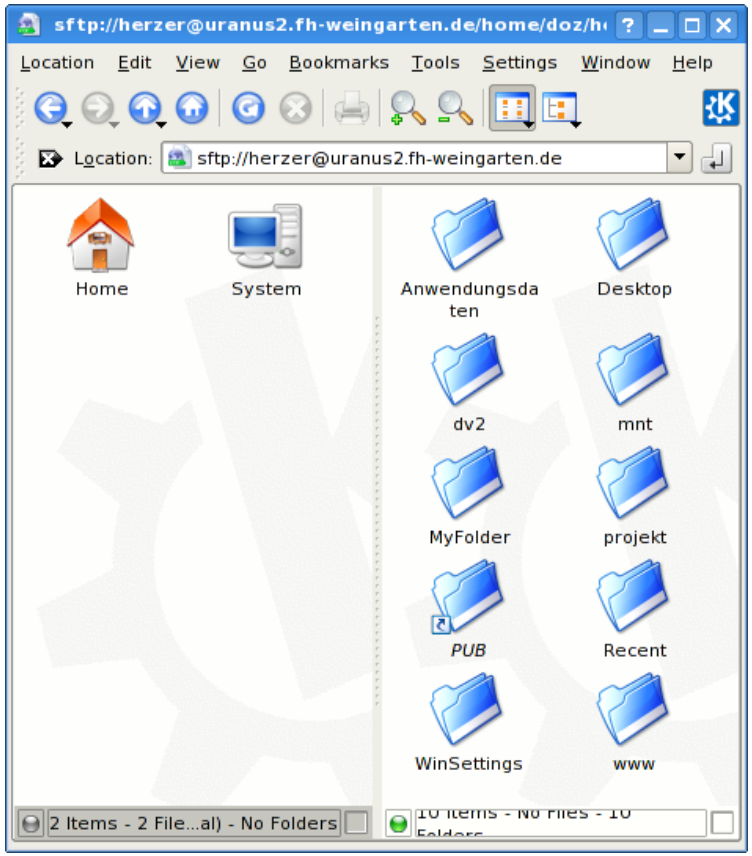
Weiterhin sollte man beachten:

- Die Felder für den `Default Gateway` und den `Primary DNS` müssen bei manchen Experimenten nicht mit Werten gefüllt sein
- Bezeichnung der Netzwerkkarte (Device-Name) merken
- Die Konfiguration kann bei Bedarf jederzeit geändert werden



Mit dem `Konqueror` kann eine verschlüsselte Verbindung zum `/home`-Verzeichnis an der Hochschule aufgebaut werden:

- `Konqueror starten`
- Ansicht in linke und rechte Hälfte teilen (`Window`→`Split View Left/Right`)
- In der linke Hälfte z.B. auf das `Home`-Symbol klicken (das ist dann lokal)
- Rechte Hälfte aktivieren
- In die Adresszeile folgendes reinschreiben



`sftp://<loginname>@uranus2.fh-weingarten.de`

- Nach bestätigen mit `Enter` das Passwort eingeben

Netzwerkkarte von Hand in einer Shell konfigurieren

`ifconfig␣eth0␣down`↵

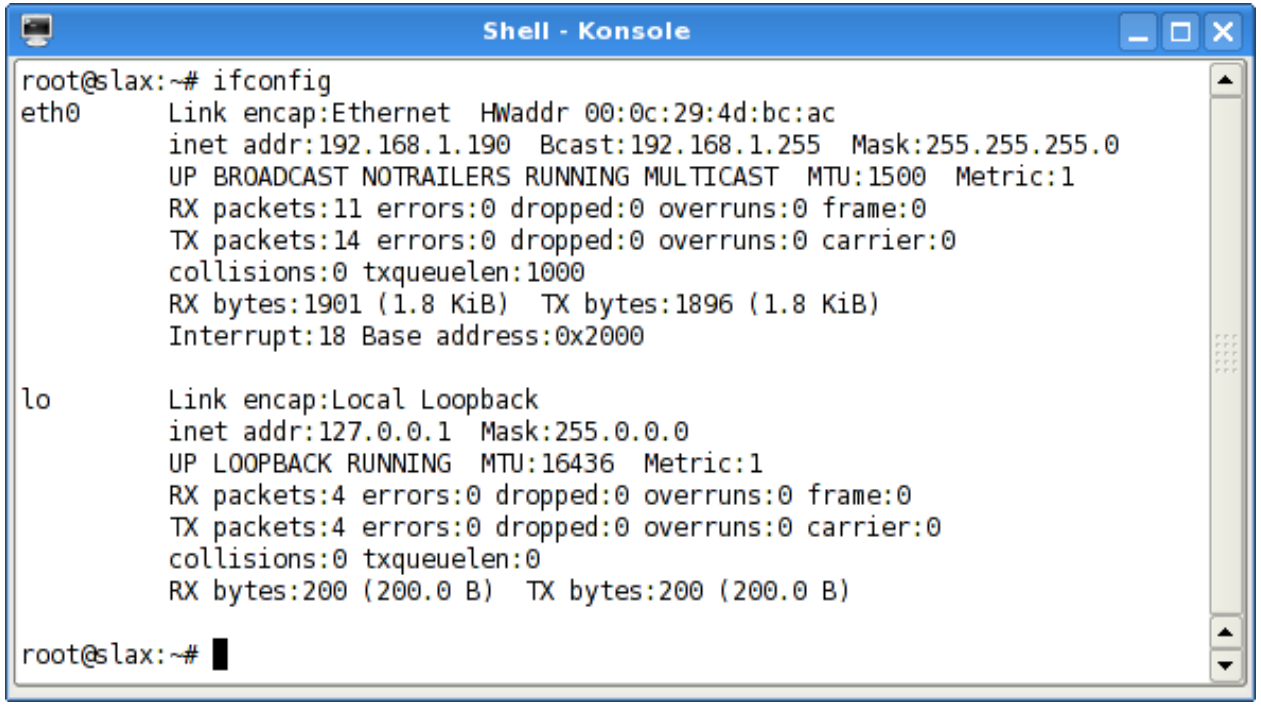
`ifconfig␣eth0␣`*<IP-Adresse>*`␣netmask␣`*<netmask>*↵

`ifconfig␣eth0␣broadcast␣`*<Broadcast-Adresse>*↵

`route␣add␣-net␣`*<IP-Adresse>*`␣netmask␣`*<netmask>*`␣dev␣eth0`↵

`route␣add␣default␣gw␣`*<default gateway>*↵

`ifconfig␣eth0␣up`↵



Beschreibung

Serverfunktionen eines Unix-Systems werden häufig über den `inetd` abgewickelt. Die Aufgabe dieses Programms ist es, vereinfacht ausgedrückt, ständig "zu lauschen" ob ein Dienst (ein Port) angefordert wird. Wenn das der Fall ist, startet der `inetd` die in `/etc/inetd.conf` eingestellten Programme (meist ebenfalls Daemonen, also Programme die mit einem `d` enden).

inetd starten bzw. nach Konfigurationsänderungen neu starten

`/etc/init.d/openbsd-inetd␣start`↵

`/etc/init.d/openbsd-inetd␣restart`↵

Besonderheiten von Slabix

Es gibt nur den User `root`

Das root-Passwort ist `toor`

/root/slabix-init enthält einige Skripte mit denen Netzwerkdienste gestartet werden können

Beschreibung

Unter einem C-Compiler versteht man eine Sammlung von Programmen, die aus einem Quellcode ein ausführbares Programm erzeugen.

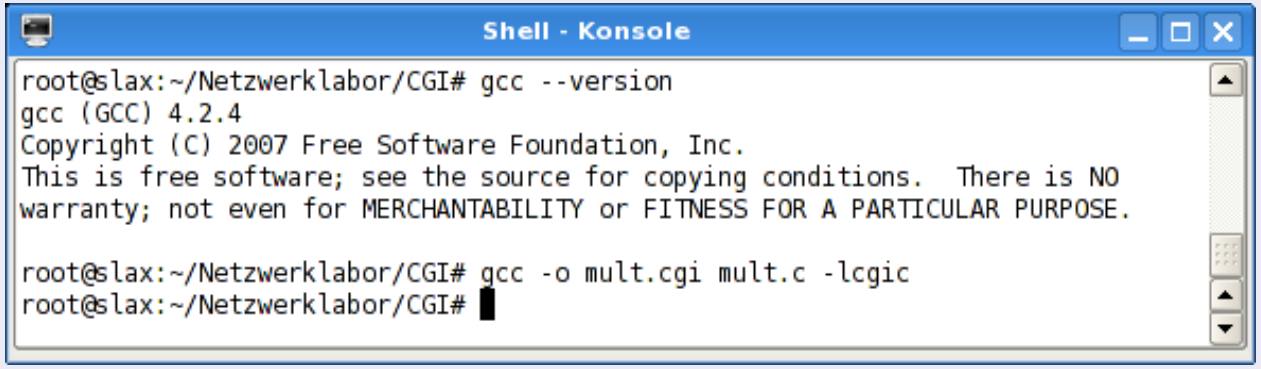
gcc

Auf der DVD ist der gcc enthalten. Mit ihm können C und C++ Quellcodes übersetzt werden.

Einige Parameter des C-Compilers

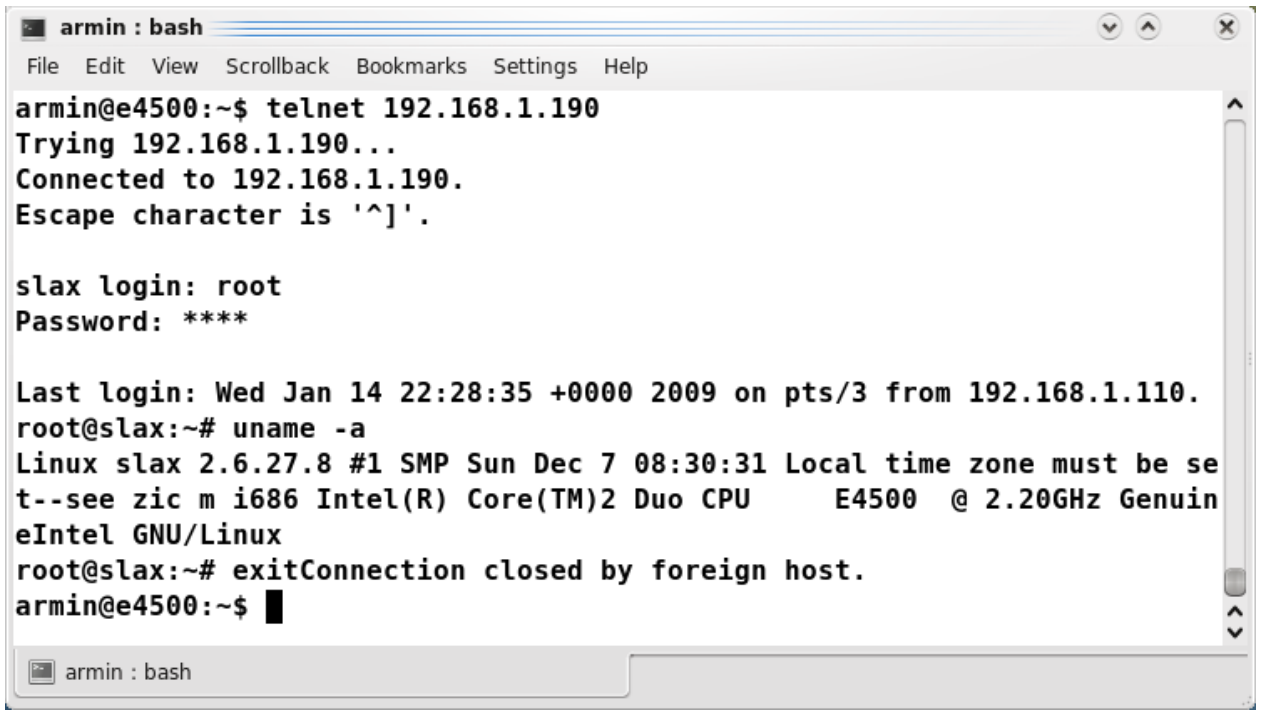
`-o␣`<*Name der ausführbaren Datei nach dem Übersetzen*↵

`-l`<*Name der Link-Library*↵



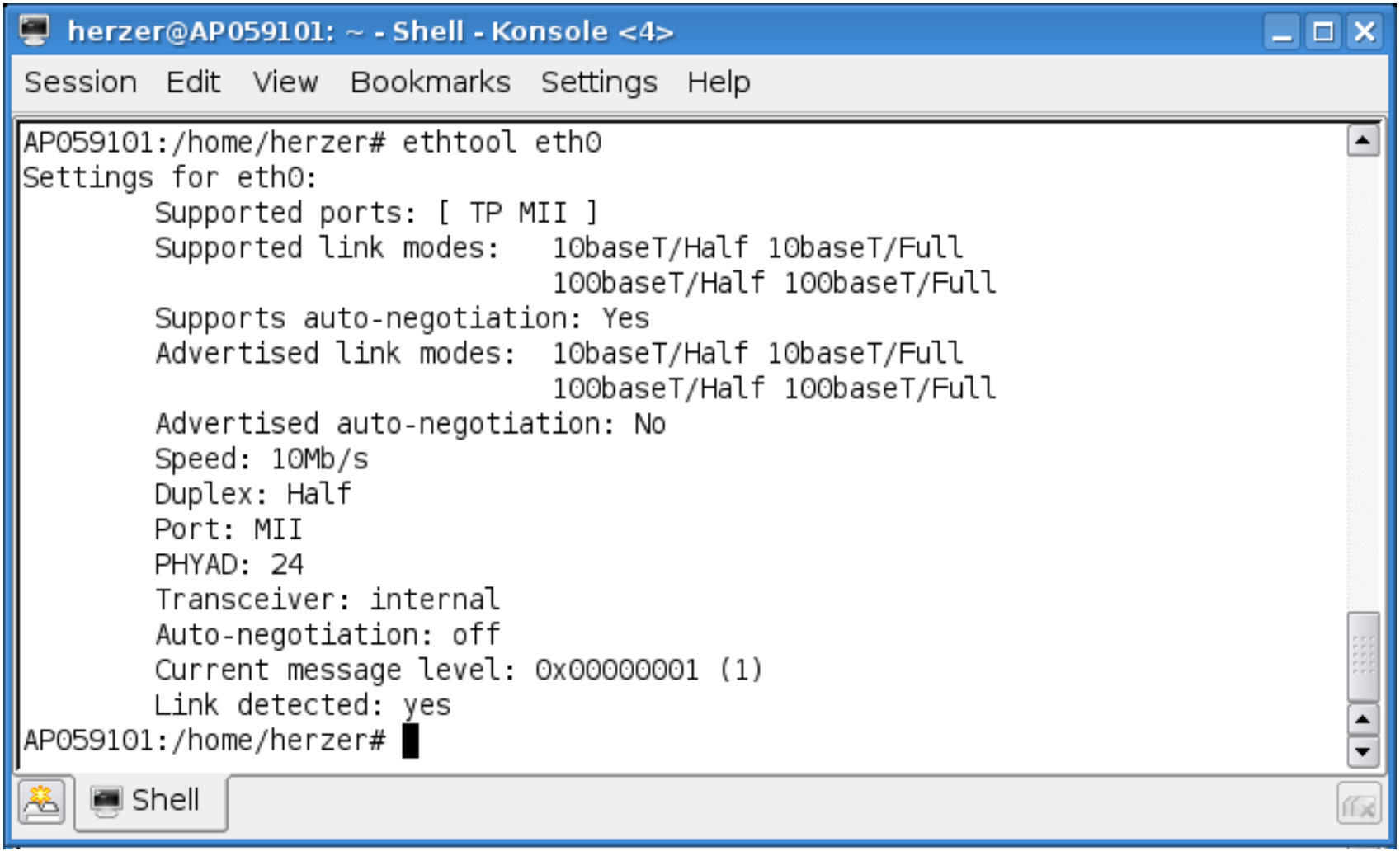
Beschreibung

Der Klassiker unter den remote connectivity Tools. Heutzutage veraltet, weil der gesamte Verkehr (inkl. Benutzername und Passwörtern) im Klartext abläuft und deshalb leicht auszuspionieren ist. Trotzdem bieten noch viele Netzwerkgeräte einen telnet-Zugang an, den man **innerhalb** aber aus o.g. Gründen nur innerhalb eines LAN'S (wenn überhaupt) nutzen sollte. Im Gegensatz zu ssh verhindern die meisten telnet daemons (per default Einstellung) daß sich ein remote user als root anmeldet!



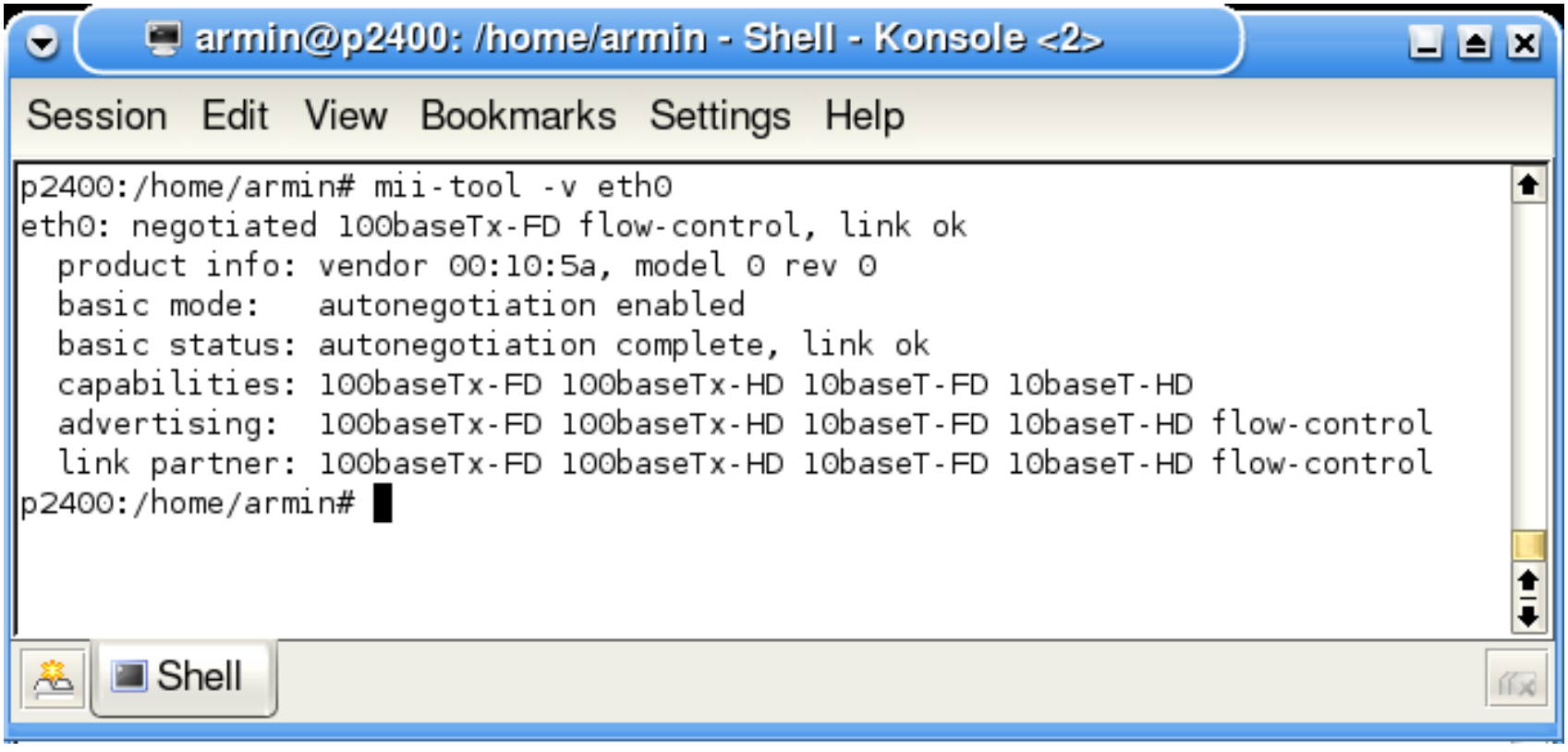
Schicht(en)	Software
7 ... 5	Anwendungen: wireshark, telnet, putty, ssh, thttpd, ... Protokolle: snmp, samba, nfs, ...
4	nmap
3	traceroute, arp, ifconfig, ...
2	wakeonlan
1	ethtool, mii-tool

Die Tabelle gibt einen kurzen Überblick über Programme und Protokolle im Kontext des OSI-Schichtenmodells. Man beachte vor allem, daß auf L5-L7 die Grenzen oft fließend sind!



Aufgabe 1 (Screenshot)

1. Betriebszustand der Netzwerkkarte abfragen und neu setzen
2. Link-LED blinken lassen. Was bringt das???



- als Alternative/Ergänzung zu `ethtool` verwendbar

Aufgabe 2 (mit `ethtool` und Screenshot/Bild dokumentieren):

1. Welche Eigenschaften unterstützt die Netzwerkkarte?
2. Was ändert sich am Switch/Hub, wenn die Geschwindigkeit von 100 auf 10 Mbps geändert wird?

Beschreibung

- Rechner über ein Netzwerk einschalten:
  - entsprechende BIOS-Funktion aktivieren
  - mit `ethttol` WOL-Funktionen überprüfen
  - "Magic Packet" schicken
- Fast alle moderne Rechner werden unterstützt
- Externe Netzwerkkarten sind oft auch WOL-fähig

Aufgabe 3 (mit Screenshot dokumentieren):

1. Mit `wireshark` ein "Magic Packet" einfangen
2. Wie lauten die Quell- und Ziel-MAC-Adressen?
3. Wie lauten die Quell- und Ziel-IP-Adressen?
4. Welcher Zielport wird mit welchem Protokoll angesprochen?
5. Wie ist der "Payload"-Teil strukturiert?

http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf

Beschreibung

- Netzwerkschnittstellen beim booten initialisieren bzw. nachträglich konfigurieren
  - IP-Adresse, Netzwerkmaske, Default Gateway...
  - MAC-Adresse
- Konfiguration überprüfen
- Fehlersuche

Aufgabe 4 (mit Screenshot dokumentieren):

1. Die Konfiguration der Netzwerkschnittstelle soll überprüft werden
2. Die MAC-Adresse soll auf 00:11:22:33:44:55 geändert und das Ergebnis überprüft werden

http://www.fibel.org/linux/lfo-0.6.0/node476.html

Beschreibung

- gibt den Inhalt des arp-Caches
- kann den arp-Cache verändern (z.B. Einträge löschen)

Aufgabe 5 (mit Screenshot bzw. Text dokumentieren)

1. Rechner anpingen und mit `wireshark` arp-Frames aufzeichnen
2. arp request und arp reply suchen und Lösungsblatt ausfüllen

http://www.oreilly.de/german/freebooks/linag2/netz0510.htm

Nmap

- konsolenbasierter Portscanner (“Network Mapper”)
- Netzwerkanalyse
  - welche Rechner sind online
  - welche Dienste (offene Ports) bieten diese Rechner an
  - welches Betriebssystem ist installiert
  - mit Zenmap steht eine grafische Oberfläche zur Verfügung
- **Nur im privaten LAN damit rumspielen (kann im Betrieb ein Kündigungsgrund sein!!!)**

Aufgabe 6 (Screenshot und Tabelle)

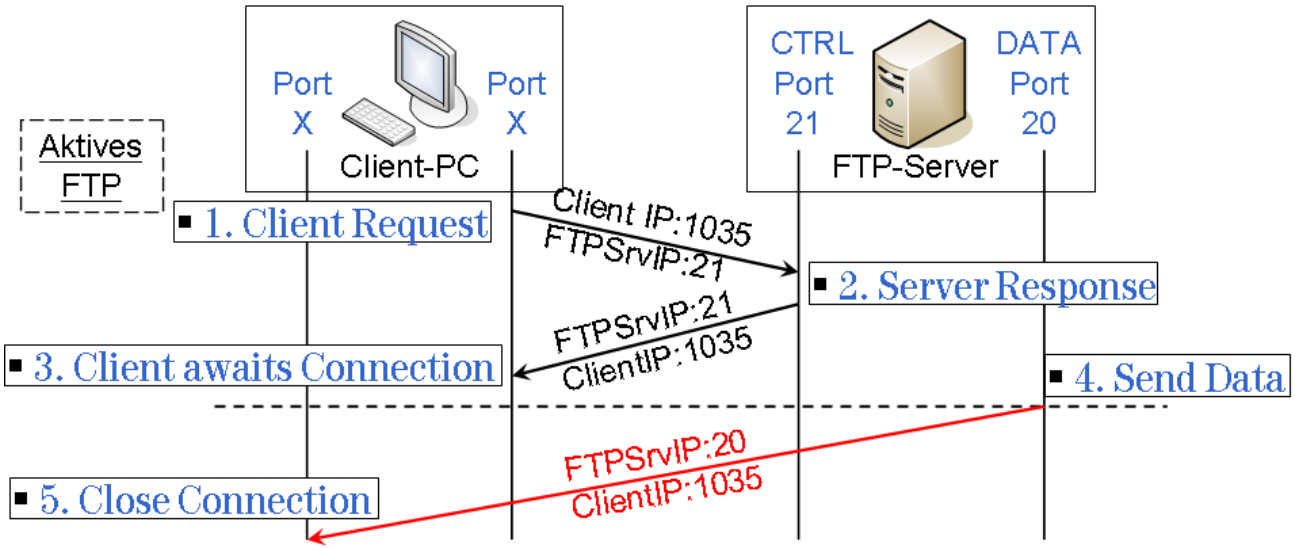
1. Mit `nmap -A -T4 192.168.1.254` wird ein Portscan über den Printserver gemacht. Welche Dienste bietet er an?
2. Mit `nmap -sP 192.168.1.0/24` erzeugt man eine Liste mit allen MAC- und IP-Adressen der Rechner im Netzwerk

http://nmap.org/

http://www.nwlab.net/tutorials/portscanner/nmap-tutorial.html

ftp

- ein in RFC 959 von 1985 definiertes Netzwerkprotokoll zur Übertragung von Daten in TCP/IP-Netzwerken
- Client/Server-Struktur
- Datenübertragung unverschlüsselt



Aufgabe 7 (mit Screenshot dokumentieren)

1. Mit `wireshark` soll der im Bild dargestellte Kommunikationsablauf mitgeschnitten werden

arping

- findet heraus ob eine IP-Adresse vergeben ist und welche MAC-Adresse zu ihr gehört (ähnlich wie `ping`)
- Beispiele:
  1. Um sich eine IP-Adresse in einem Netzwerk, von dem man nicht weiß, welche IP-Adressen bereits benutzt werden, zu vergeben müßte man auf gut Glück IP-Adressen durchprobieren oder man verwendet `arping` um eine Adresse zu testen.
  2. MAC-Adresse anpingen (z.B. `arping2 00:11:22:33:44:55`)
  3. IP- und MAC-Adresse gleichzeitig anzeigen lassen
  4. Funktioniert auch z.B. bei Windows XP-Rechnern, die ICMP-Pakete (also "richtige" pings) normalerweise blocken

Aufgabe 8 (Screenshots)

- Das unter 1 beschriebene Szenario durchführen

arpoison

- ARP ist ein zustandsloses Prototkoll (das bedeutet, es werden auch ARP-Replies bearbeitet, die nicht angefordert wurden)
- ARP authentifiziert nicht (das heißt, den Informationen der ARP-Replies wird blind vertraut)
- **Problem:** ARP ist anfällig für DOS[a]- bzw. Sniffing-Attacken wie z.B. MAC-Flooding oder ARP-Spoofing

[a] Denial of Service

Aufgabe 9 (Screenshots)

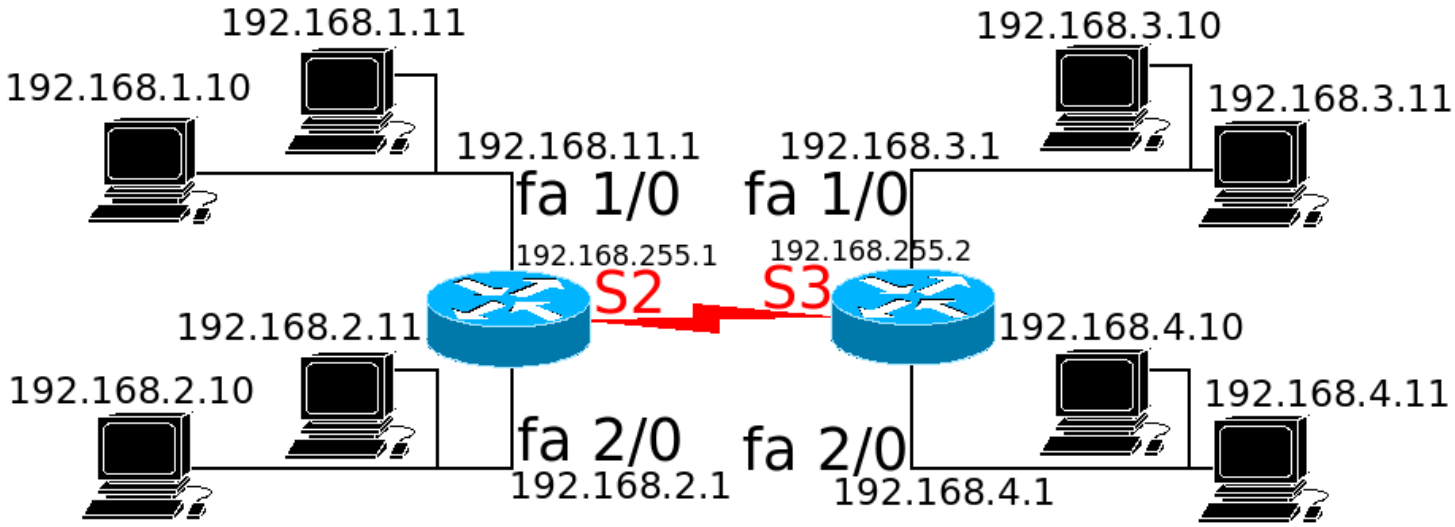
Achtung: Die nachfolgende Technik darf, außer zu genehmigten Testzwecken nicht in "produktiven" Umgebungen angewendet werden. Arbeitsrechtliche Konsequenzen könnten die Folge sein!

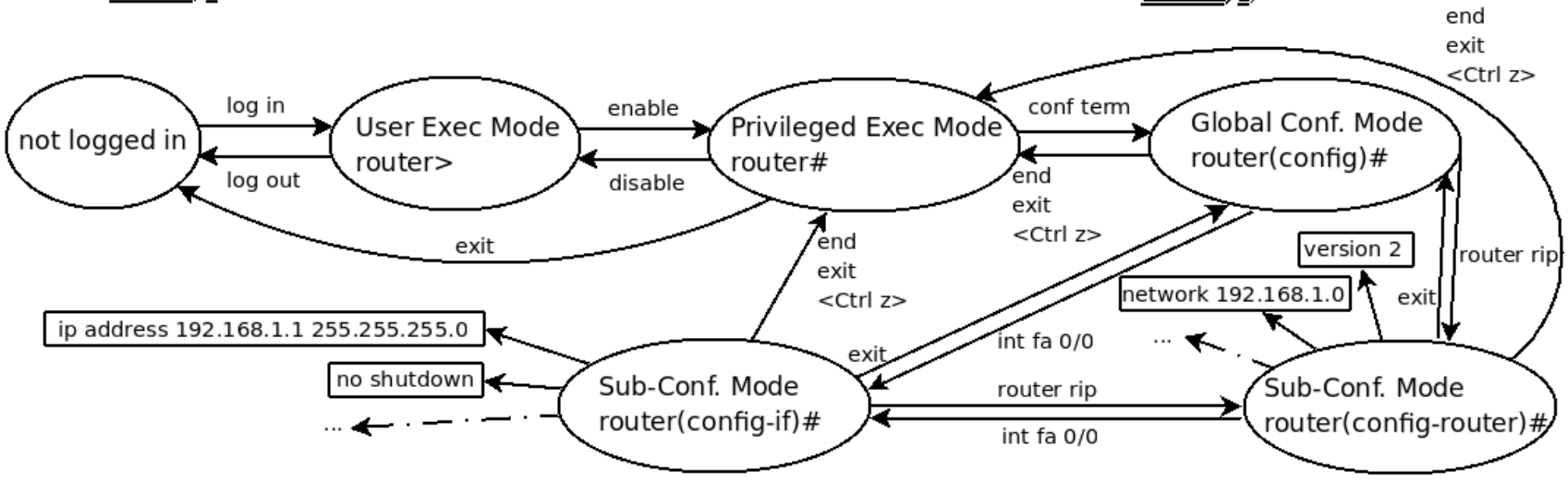
- Es soll ein DOS-Angriff auf einen Switch durchgeführt werden

Aufgabe 10 (Screenshots)

Alle Rechner und der Server werden zunächst an einen Hub (Aufgaben a. und b.) und danach an einen Switch (Aufgaben c. und d.) angeschlossen und für das Netzwerk 192.168.10/24 konfiguriert. Danach erfolgen Traffic-Analysen mit Wireshark:

a. Hub und telnet
b. Hub und ssh
c. Switch und telnet
d. Switch und ssh





Aufgabe 11 (Tabelle erstellen)

- Routing-Tabellen der beiden Router ermitteln.

Aufgabe 12

Einer Firma steht für ihr Intranet der Bereich 128.10.240.0 bis 128.10.247.255 aus dem IP-Adressraum zur Verfügung.

a. Wieviel IP - Adressen enthält dieser Adressbereich?
b. Zur Adressierung von Teilnetzen wird das Verfahren Subnetting verwendet. Die Rechneradressen (hostid) sollen dabei ein Länge von 6 Bit haben.
   b1. Geben sie eine Subnetzmaske in dezimaler Form an, mit der eine solche Adressierung realisiert werden kann.
   b2. Wieviele Rechner können pro Teilnetz adressiert werden? Berücksichtigen Sie, daß bestimmte Adressen u.U. reserviert sein können.
   b3. Wieviele Teilnetze können adressiert werden? Berücksichtigen Sie, daß bestimmte Adressen u.U. reserviert sein können.
   b4. Teilen Sie unter Berücksichtigung der gewählten Subnetzmaske die IP - Adresse 128.10.240.68 in netid, subnetid und hostid auf. Geben sie das Ergebnis in dezimaler Form an.

Aufgabe 13

a. Das Netz 192.168.0.0 soll in vier Subnetze geteilt werden. Ermitteln Sie die notwendige Netzmaske und schreiben Sie alle Netze mit ihrer Ober- und Untergrenze auf.

b. Das Netz 20.0.0.0 soll in 121.792 Subnetze geteilt werden. Ermitteln Sie die erforderliche Netzmaske und schreiben Sie die Netze Nummer 1, 2, 10000 und das letzte Netz mit ihrer Ober- und Untergrenze auf.

c. Für das Netz 11.0.0.0 soll eine Einteilung gewählt werden, so daß für 1.789 PC eine eindeutige IP-Adresse vergeben werden kann und möglichst wenige Adressen verschwendet werden.

Aufgabe 14

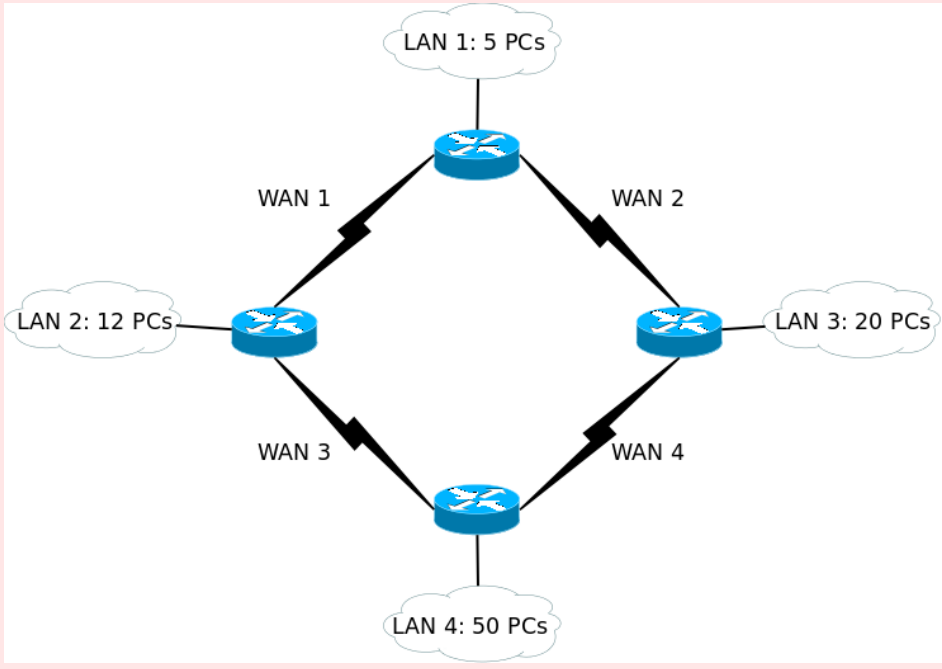
Ein Netzwerk mit der Netzwerkadresse 201.160.15.0/24 soll folgendermaßen aufgeteilt werden:

LAN1 : 5 PCs

LAN2 : 12 PCs

LAN3 : 20 PCs

LAN4 : 50 PCs



Die LANs sollen mit 4 WANs untereinander vernetzt sein!

a. Ermittle die Anzahl der erforderlichen Subnetze.

b. Berechne die IP's, Einstellungen, Adressen.

Aufgabe 15

Ein Unternehmen möchte ein strukturiertes Computernetzwerk aufbauen. In einem Bürohaus befinden sich mehrere Räume, in die sich das Unternehmen eingemietet hat. Organisatorisch gibt es mehrere Arbeitsgruppen, die völlig unabhängig voneinander arbeiten. Der Netzwerkadministrator empfiehlt deshalb den Aufbau von entsprechend vielen Subnetzen mit folgenden Daten: Das Netzwerk 192.168.130.0 wird über die Subnetzmaske 255.255.255.224 aufgeteilt.

a. Wie viele Arbeitsgruppen kann es damit maximal geben?

b. Wie viele Hosts darf eine Arbeitsgruppe maximal haben?

c. Erstellen Sie (z.B. mit OOCalc) eine Tabelle aller Subnetze mit folgendem Inhalt:

   c1. Subnetznummer

   c2. Adressbereiche für jedes Subnetz

   c3. Broadcastadressen für jedes Subnetz.

d. In welchem Subnetz sind die folgenden IP-Adresse:

d1. 192.168.130.96	d2. 192.168.130.67	d3. 192.168.130.93
d4. 192.168.130.199	d5. 192.168.130.222	d6. 192.168.130.255

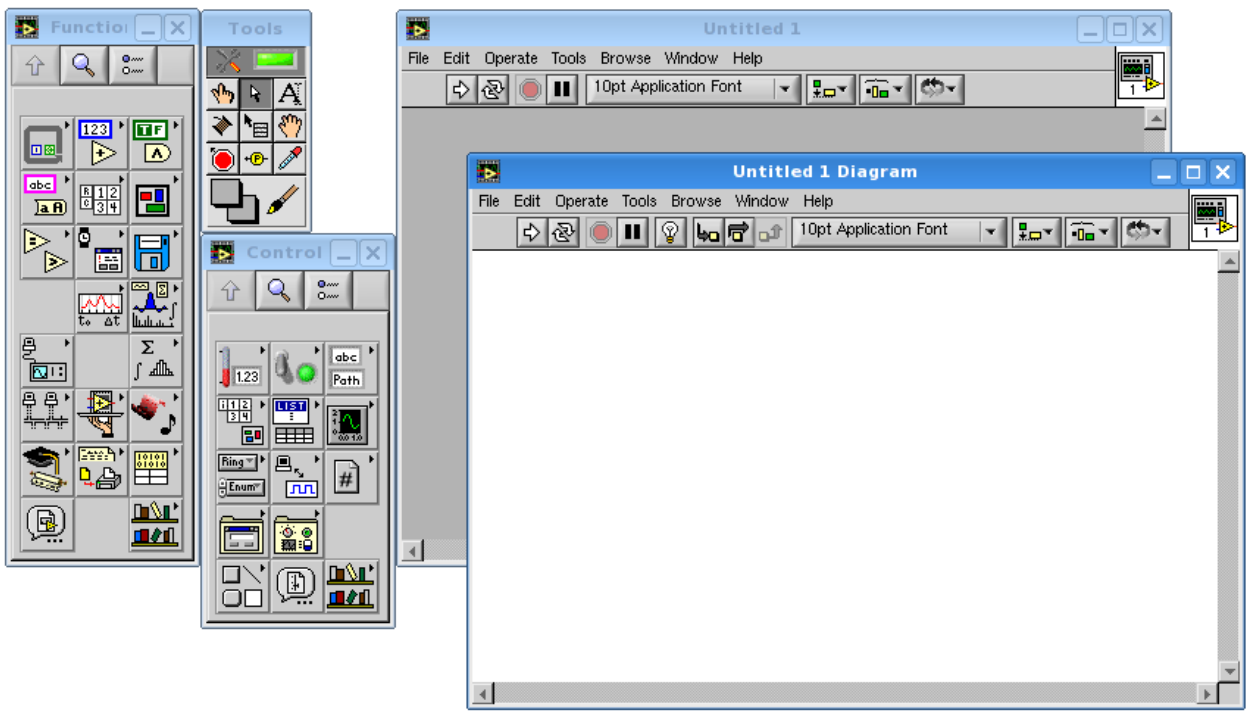
In diesem Experiment sollen Begriffe wie

- Protokollarten TCP und UDP
- Portnummern
- Client und Server

anhand einfacher Beispiel in den Programmiersprachen C und LabVIEW (`http://www.natinst.de`) untersucht werden.

Aufgabe 16

LabVIEW aus dem K-Menu→Development starten



Pfeilwerkzeug: Universelles Werkzeug zum platzieren, markieren, verschieben oder auswählen aller möglichen Elemente

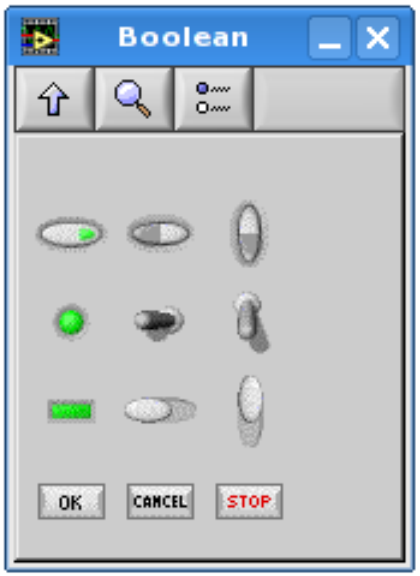
Eingabewerkzeug: Damit können Elemente ausgewählt und editiert werden

Textwerkzeug: Wird zum eingeben oder ändern von Zahlen und Texten in Eingabefeldern und Labels

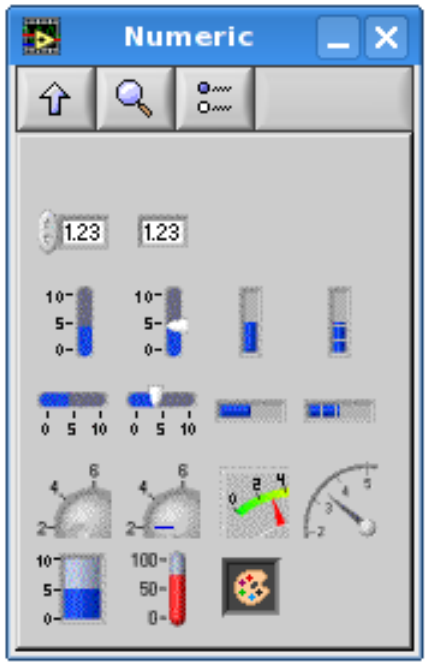
Drahtrolle: Dient zum Verknüpfen von Programmierlementen im Block Diagram

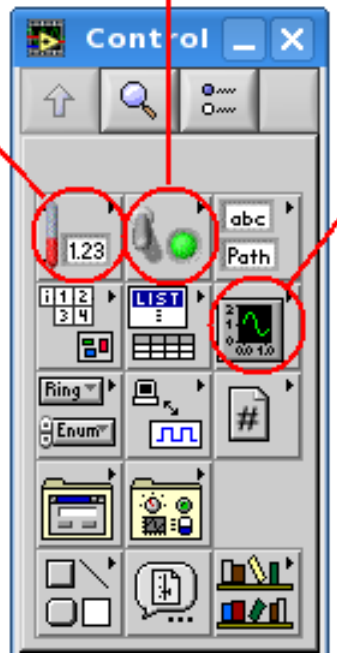
Scrollwerkzeug: Damit kann man Rollbalken bewegen

Schalter und Lämpchen

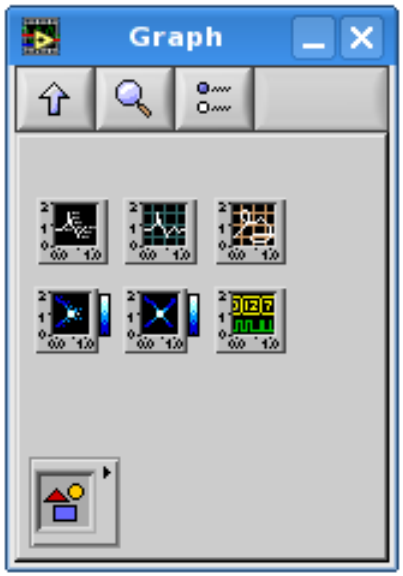


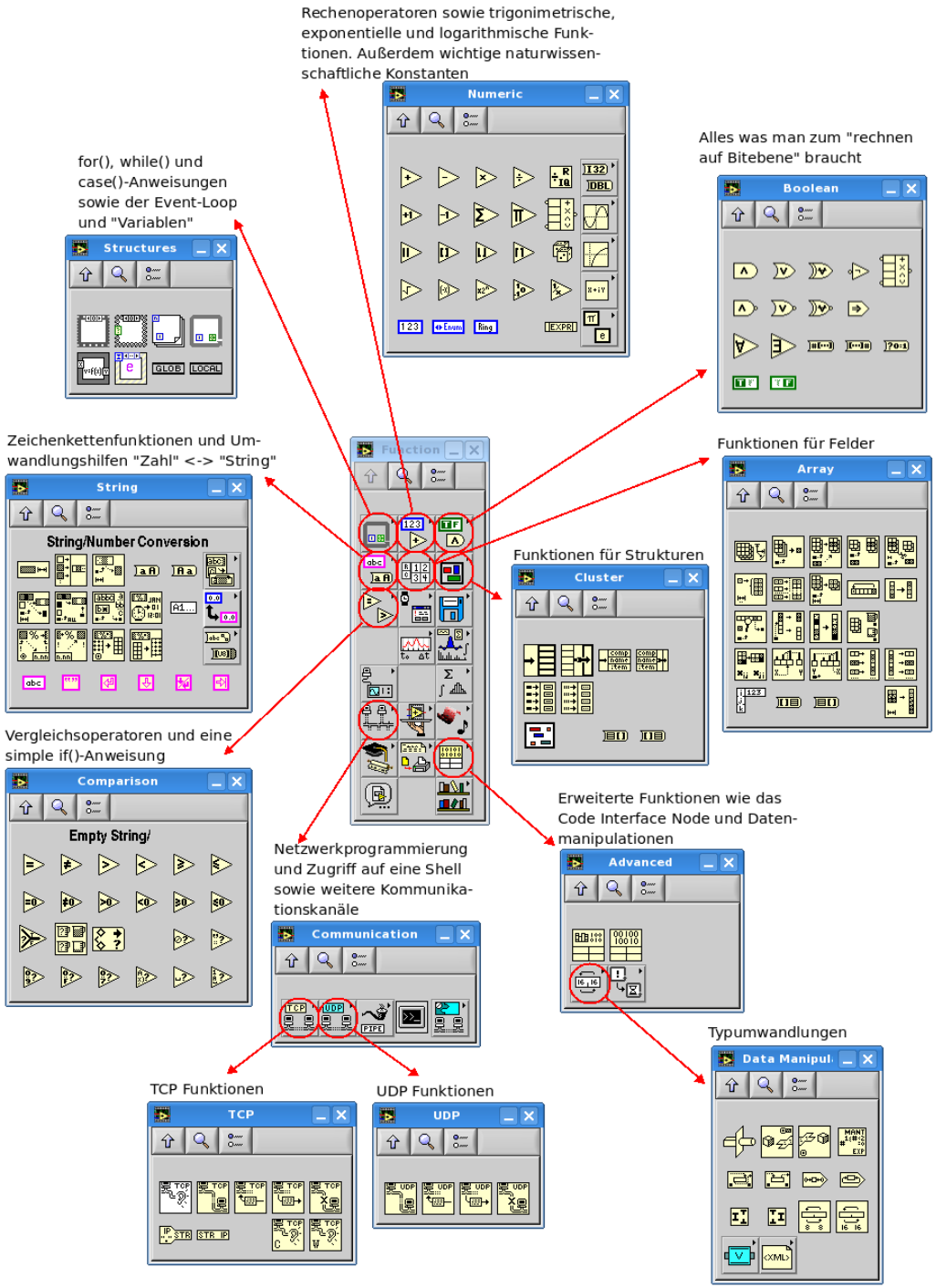
Eingabefelder, Schiebe- und Drehregler



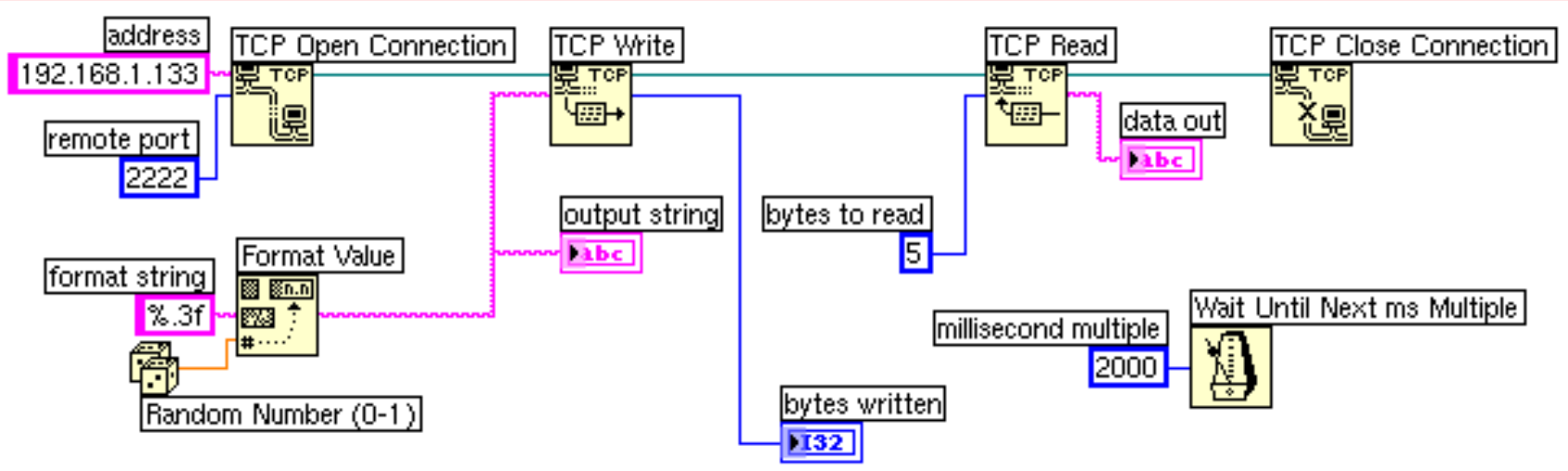


Verschiedene Diagrammtypen




Rechenoperatoren sowie trigonimetrische, exponentielle und logarithmische Funktionen. Außerdem wichtige naturwissenschaftliche Konstanten
Numeric
for(), while() und case()-Anweisungen sowie der Event-Loop und "Variablen"
Structures
Alles was man zum "rechnen auf Bitebene" braucht
Boolean
Zeichenkettenfunktionen und Umwandlungshilfen "Zahl" <-> "String"
String
String/Number Conversion
Function
Funktionen für Felder
Array
Funktionen für Strukturen
Cluster
Vergleichsoperatoren und eine simple if()-Anweisung
Comparison
Empty String/
Erweiterte Funktionen wie das Code Interface Node und Datenmanipulationen
Advanced
Netzwerkprogrammierung und Zugriff auf eine Shell sowie weitere Kommunikationskanäle
Communication
Typumwandlungen
Data Manipul
TCP Funktionen
TCP
UDP Funktionen
UDP

Aufgabe 17



a. Erstellen Sie auf dem Client-Rechner das o.a. LabVIEW-Programm

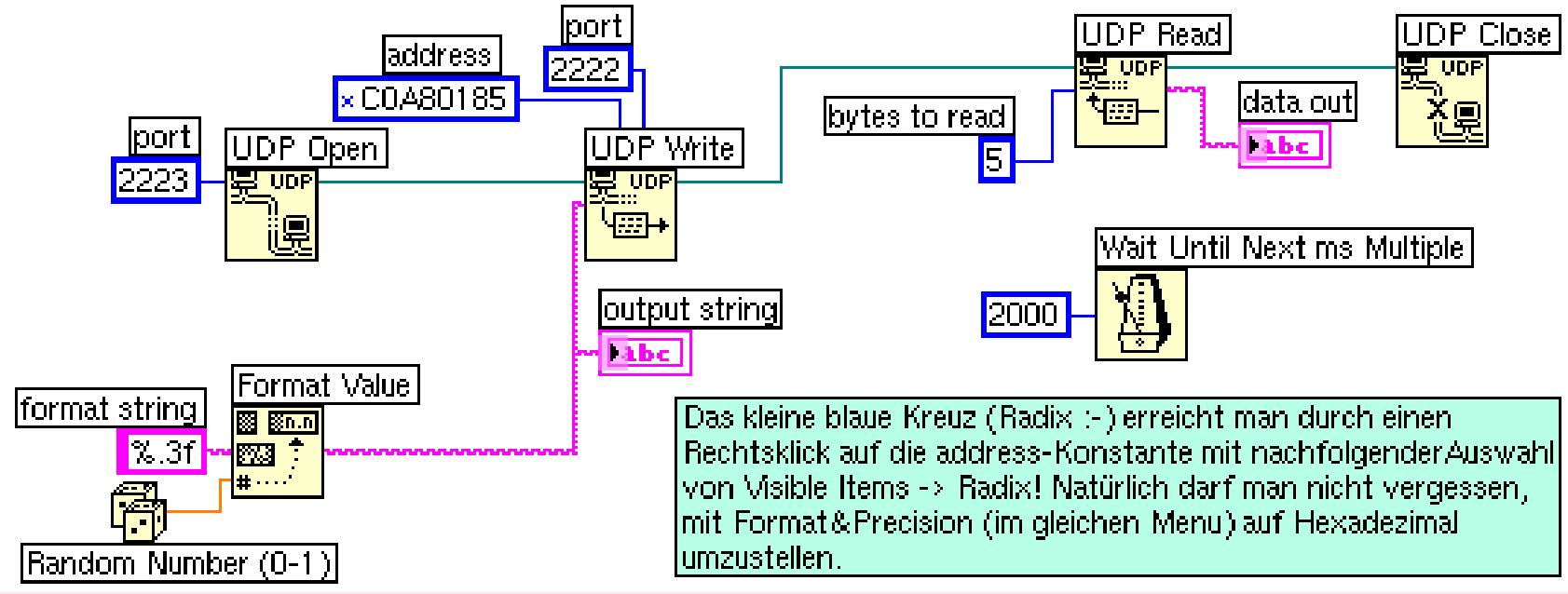
b. Auf dem anderen Rechner den TCP-Server starten: `tcpserver1␣2222`↵

c. Das LabVIEW-Programm mit dem weißen `Run`-Pfeil in der Steuerleiste starten. Dokumentieren Sie die jeweiligen Ausgaben im LabVIEW-Frontpanel und im Konsolen-Fenster des Servers mit einem Screenshot.

d. Das LabVIEW-Programm mit der `Run Continuously`-Schaltfläche starten. Screenshot des Konsolefensters!

e. Entfernen Sie die "Zeitbremse"! Dann starten Sie das Programm wieder im `Run Continuously`-Modus. Was beobachten Sie nun?

Aufgabe 18

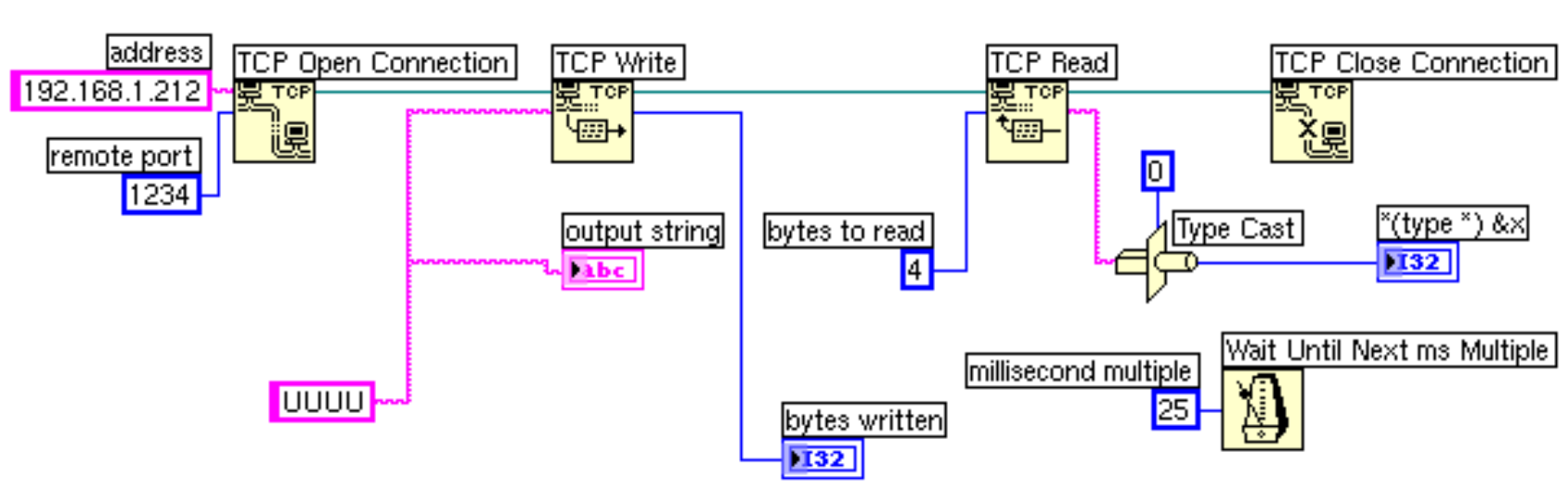


a. Erstellen Sie auf dem Client-Rechner das o.a. LabVIEW-Programm

b. Auf dem anderen Rechner den UDP-Server starten: `udpserver1␣2222`↵

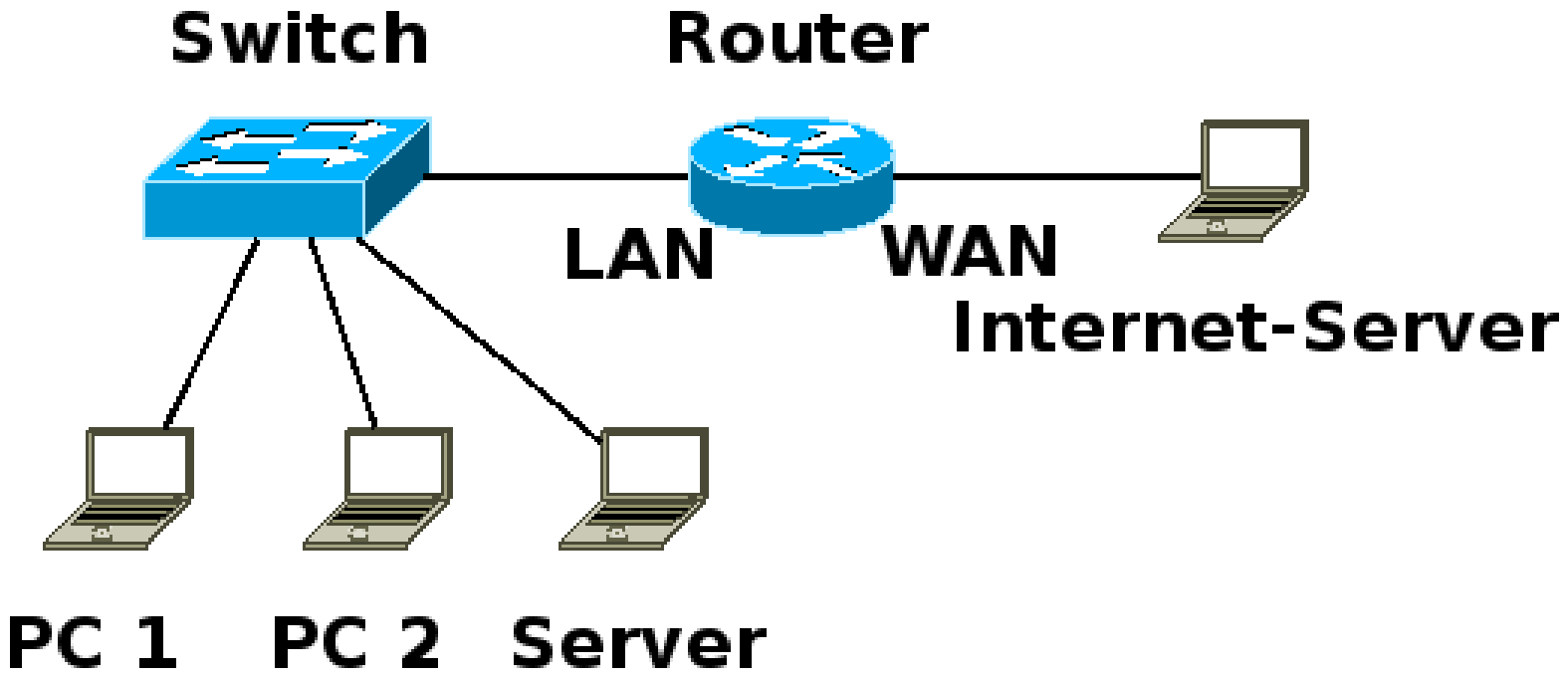
e. Vergleichen Sie den `wireshark`-Mitschnitt mit dem vom TCP-Server. Welche Unterschiede fallen auf?

Aufgabe 19



a. Es soll ein LabVIEW-Programm erstellt werden, mit dem über einen Longshine Deviceserver LCS-C844 Zufallszahlen aus eine TRNG[a] abgeholt werden können. Hinweis: Es handelt sich dabei um einen geringfügig modifizierten TCP-Client von Folie 44).

[a] True Random Number Generator



Übersicht

- Es bilden sich Paare aus Client-/Server-Rechnern
- Auf den Server-Rechnern wird ein System in Betrieb genommen, mit dem gemessen werden kann, wieviele Bytes die WAN-Schnittstelle des Routers in Richtung LAN bzw. WAN passiert haben. Das Ergebnis soll über einen Webserver kommuniziert werden.
- Auf den Client-Rechner kann das Ergebnis der Überwachung in einem Web-Browser dargestellt werden.

Aufgabe 20

a. Abfragen der SNMP-Eigenschaften:

```
snmpwalk -c public -v 1 192.168.1.1 enterprises ↵
```

b. **enterprises** durch `ip` bzw `interfaces` ersetzen

c. Ausgabe in muerischer Form:

```
snmpwalk -c public -v 1 192.168.1.1 -On system ↵
```

d. Parameter gezielt abfragen:

```
snmpget -c public -v 1 192.168.1.1 sysUpTime.0 ↵
```

Aufgabe 21

a. Mit `cd /root/Netzwerklabor/snmp` in das Verzeichnis mit den vorgefertigten Skripten wechseln

b. `./create_rra.sh`

c. `cp zywall1.awk /tmp`

d. `cp zywall1.sh /tmp`

e. `./init_thttpd.sh`

f. Mit der folgenden Zeile wird ein Editor gestartet mit dem man cronjobs bearbeiten kann:

   `crontab␣-e`↵

   Wird nun die folgende Zeile in den Editor eingegeben, dann wird das Skript `zywall1.sh` automatisch pro Minute einmal ausgeführt:

   `*␣*␣*␣*␣*␣/tmp/zywall1.sh`↵

   Abspeichern nicht vergessen. Zur Sicherheit kann man den Eintrag in die cron-Tabelle mit folgendem Kommando überprüfen:

   `crontab␣-l`↵

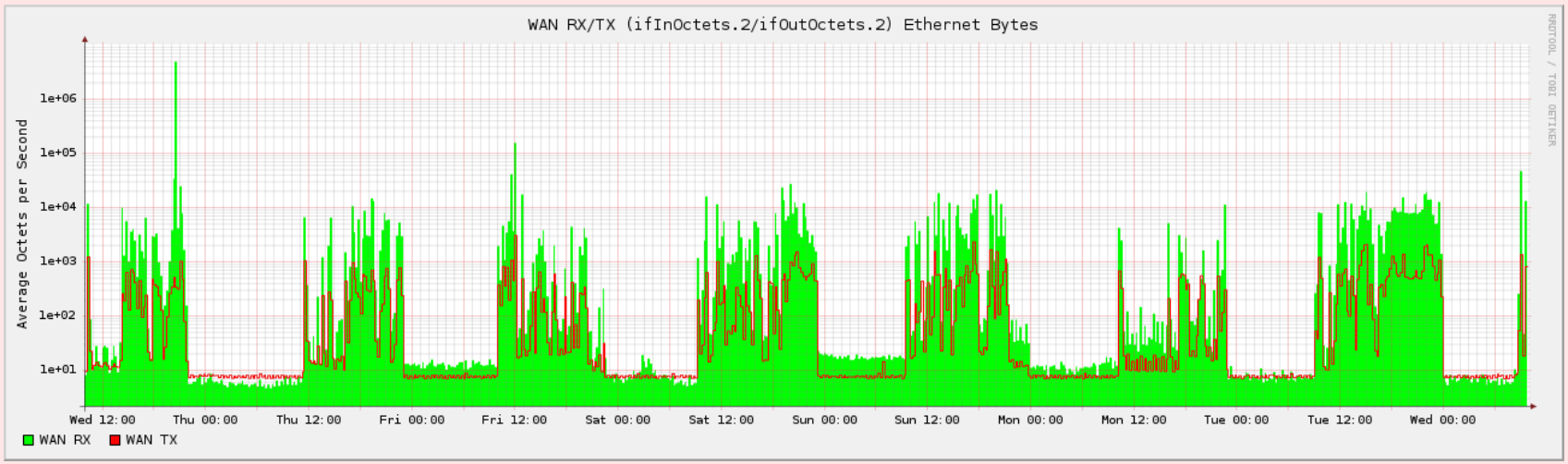
Aufgabe 22

a. Das System beginnt nun zu arbeiten und nach 3-4 Minuten kann man überprüfen ob die Datenbank auch gefüllt wird:

   `rrdtool␣fetch␣/tmp/zywall1.rrd␣AVERAGE`↵

   Wenn alles richtig konfiguriert wurde, dann kann man auf dem Server (Zugriff mit http://<IP-Adresse Server>) die Traffic -Statistik einsehen.

Die Belegung eines RJ45-Steckers und die Pinbelegung des WAN-Interfaces an einem Router, so wie SNMP sie sieht, sind hier abgebildet: In der folgenden Abbildung sieht man den Datenverkehr, aufgezeichnet über den Zeitraum einer Woche, wie man ihn erhält, wenn man das obere Verfahren anwendet und an den entsprechenden Stellen leicht modifiziert:



In diesem Experiment sollen Begriffe wie

- symmetrische und asymmetrische Verschlüsselung
- privater und öffentlicher Schlüssel
- digitale Unterschrift

mit Hilfe des OpenSSL-Toolkits[4] untersucht werden[5].

Aufgabe 23

Immer zwei Rechner bilden ein Team aus Alice und Bob.

a. Alice erstellt auf ihrem Rechner zunächst einen privaten und aus diesem einen öffentlichen Schlüssel:

```
openssl␣genrsa␣>alice.private↵
openssl␣rsa␣-pubout␣<alice.private␣>alice.public↵
```

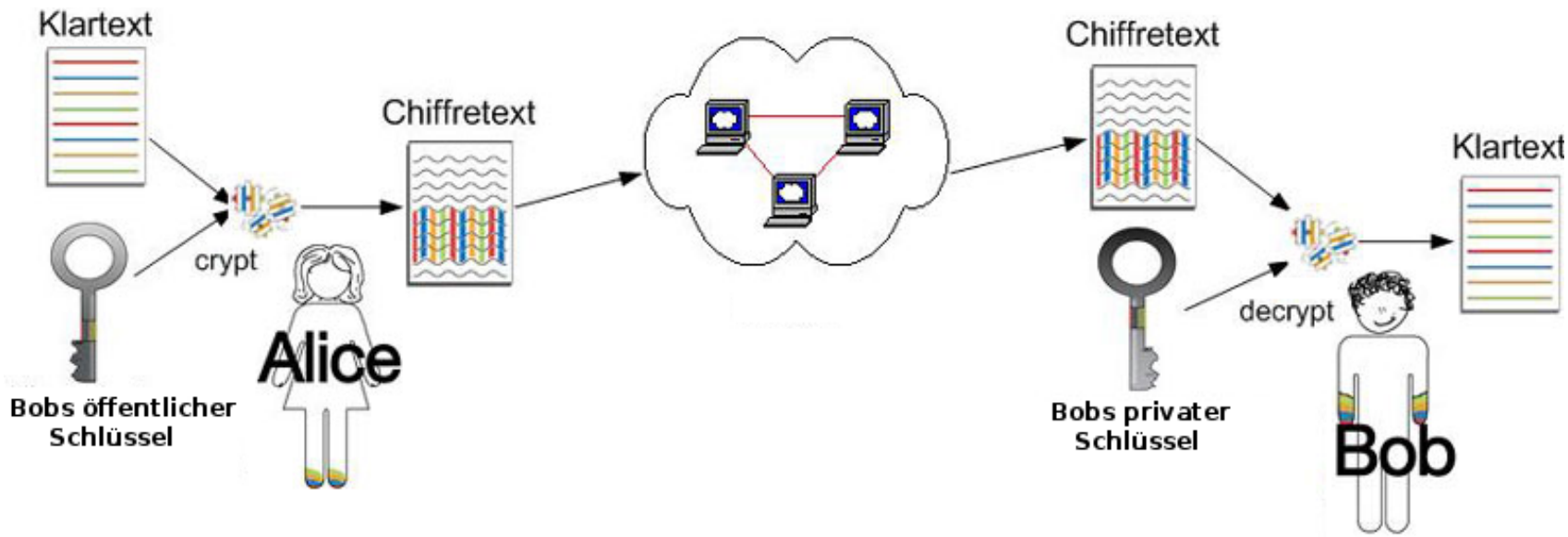
b. Bob erstellt auf seinem Rechner einen privaten und einen öffentlichen Schlüssel:

```
openssl␣genrsa␣>bob.private↵
openssl␣rsa␣-pubout␣<bob.private␣>bob.public↵
```

c. Tauschen der **öffentlichen** Schlüssel

[4] http://www.openssl.org
[5] http://www.cryptoshop.com

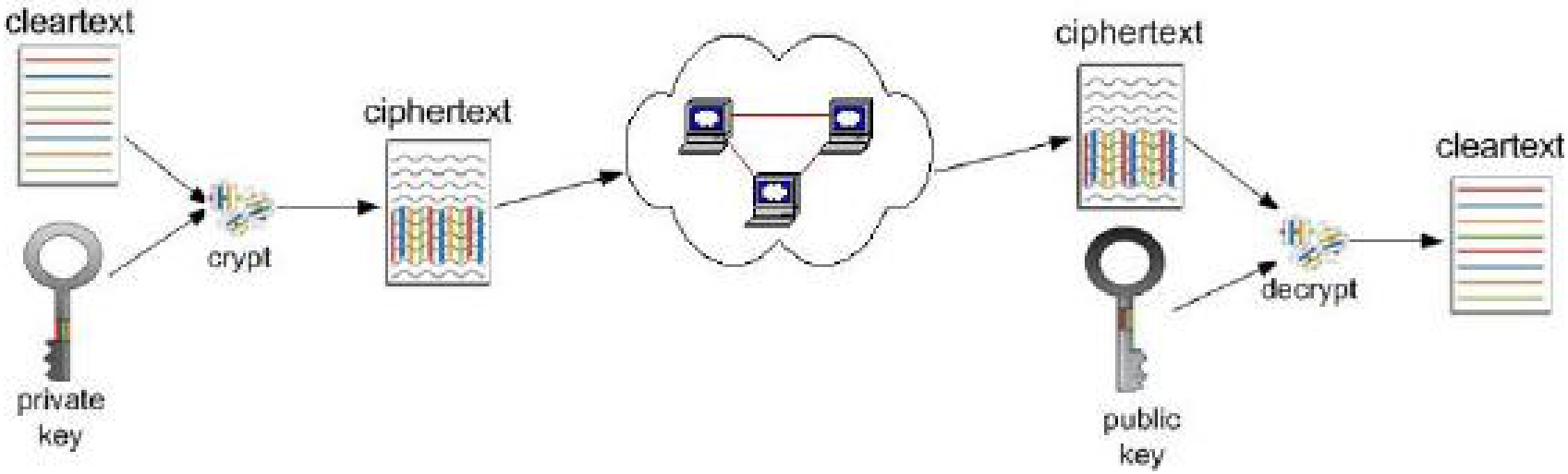
Alice kann nun mittels asymmetrischer Verschlüsselung Bob eine vertrauliche Nachricht senden, ohne dass die beiden denselben symmetrischen Schlüssel besitzen müssen. Dazu verschlüsselt Alice mit dem öffentlichen Schlüssel von Bob. Jetzt kann nur noch Bob mit seinem privaten Schlüssel die Nachricht wieder entschlüsseln. Mallory scheitert daran, da der öffentliche Schlüssel nutzlos ist. Bob weiß nun, daß niemand anderer als er und der Absender diese Nachricht kennt, **aber ist Alice tatsächlich der Absender?**




```
echo␣ ¨Alice␣loves␣you¨␣>message.plain↵
openssl␣rsautl␣-encrypt␣-in␣message.plain␣-out␣message.encrypted␣\
                                        -pubin␣-inkey␣bob.public↵
```

Für die asymmetrischen privaten bzw. öffentlichen Schlüssel von Bob und Alice gilt:

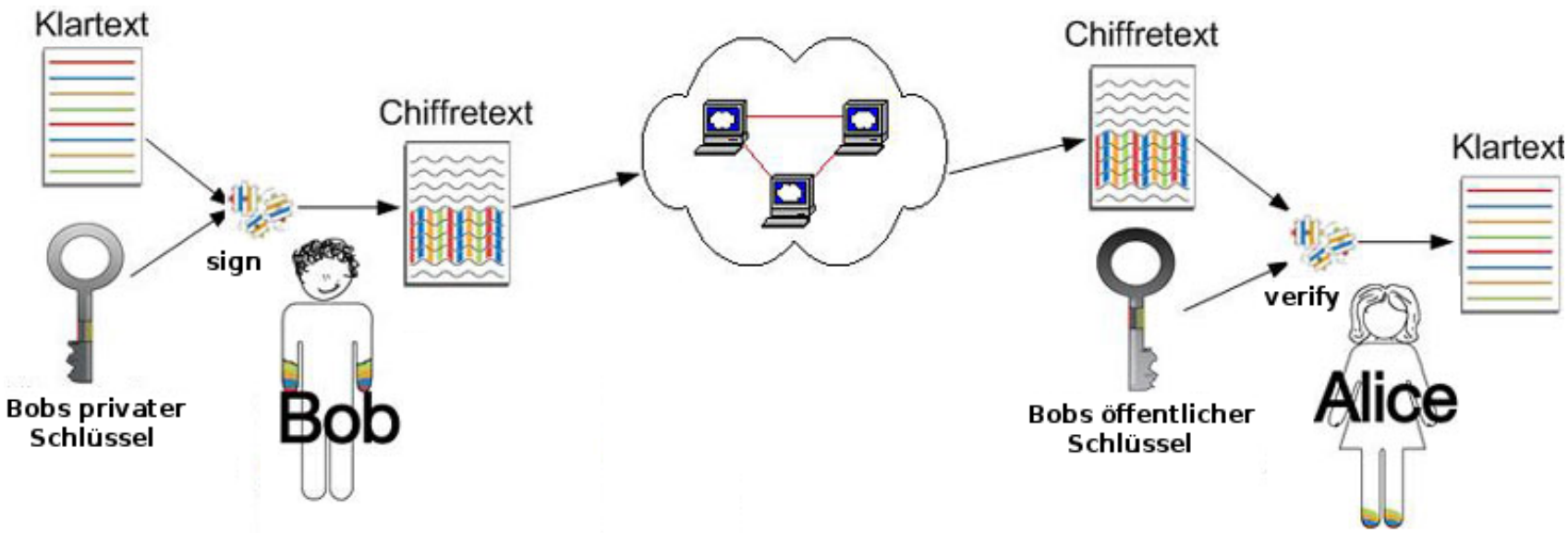
- Was mit dem privaten Schlüssel verschlüsselt wird kann nur noch mit dem öffentlichen Schlüssel entschlüsselt werden (und umgekehrt)



Bob **entschlüsselt** die Nachricht von Alice mit seinem **privaten Schlüssel**:

```
openssl␣rsautl␣-decrypt␣-in␣message.encrypted␣-out␣\
                              message.decrypted␣-inkey␣bob.private↵
```

Will Bob seine Nachricht als von ihm stammend authentisieren, wendet er seinen privaten Schlüssel auf die Nachricht an. Alice kann nun mit dem öffentlichen Schlüssel von Bob erkennen, ob die Nachricht tatsächlich von Bob stammt. Mallory kann dies zwar ebenso, schafft es aber nicht, die Nachricht unbemerkt zu verändern, da sie nicht den privaten Schlüssel von Bob besitzt.



Nun erstellt Bob eine kurze Antwort und schickt sie, mit seinem **privaten Schlüssel digital signiert**, an Alice:

```
echo␣"Will␣you␣marry␣me?"␣>message.plain↵
openssl␣rsautl␣-sign␣-in␣message.plain␣-out␣message.signed␣\
                                          -inkey␣bob.private↵
```

Alice kann mit dem öffentlichen Schlüssel von Bob sicherstellen, daß die Nachricht wirklich von Bob ist:

```
openssl␣rsautl␣-verify␣-in␣message.signed␣-out␣message.verified␣\
                                        -pubin␣-inkey␣bob.public↵
```

Nun schickt Alice Bob einen längeren Text zu. Dieser soll verschlüsselt und von Alice digital signiert werden. Alice braucht dazu mehrere Schritte:

- Einen symmetrischen Schlüssel erstellen

```
openssl␣rand␣16␣-out␣key.plain↵
```

- Mit diesem Schlüssel die Nachricht verschlüsseln

```
openssl␣des3␣-e␣-kfile␣key.plain␣-in␣message.plain␣-out␣\
                                              message.encrypted↵
```

- Eine Prüfsumme über die Nachricht erstellen

```
openssl␣dgst␣-binary␣message.plain␣>message.digest↵
```

- Mit dem privaten Schlüssel von Alice die Nachricht digital signieren

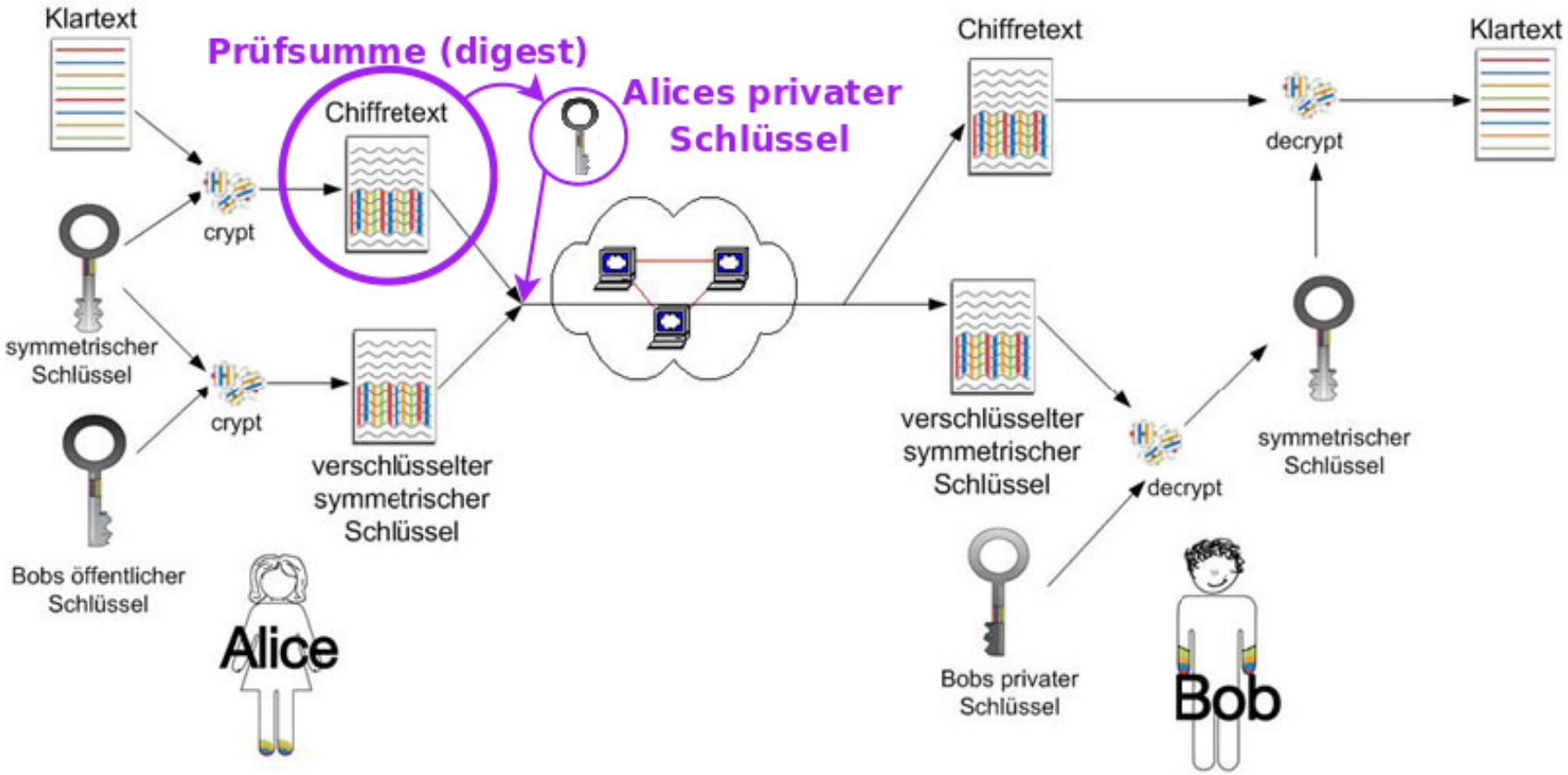
```
openssl␣rsautl␣-sign␣-in␣message.digest␣-out␣digest.signed␣\
                                          -inkey␣alice.private↵
```

- Mit dem öffentlichen Schlüssel von Bob den symmetrischen Schlüssel aus Schritt 1 verschlüsseln

```
openssl␣rsautl␣-encrypt␣-in␣key.plain␣-out␣key.encrypted␣\
                                          -pubin␣-inkey␣bob.public↵
```

Alice muß nun an Bob folgende Dateien schicken:

- die verschlüsselte Nachricht
- den verschlüsselten Schlüssel (um die Nachricht zu entschlüsseln)
- die digital unterschriebene Prüfsumme



Bob macht nach dem Empfang der Nachricht folgende Schritte:

- Mit seinem privaten Schlüssel den symmetrischen Schlüssel wiederherstellen um die Nachricht dechiffrieren zu können:

```
openssl␣rsautl␣-decrypt␣-in␣key.encrypted␣-out␣\
                                    key.decrypted␣-inkey␣bob.private↵
```

- Mit dem rückgewonnenen symmetrischen Schlüssel die Nachricht lesen

```
openssl␣des3␣-d␣-kfile␣key.decrypted␣-in␣message.encrypted␣\
                                    -out␣message.decrypted↵
```

- Bob überprüft mit dem öffentlichen Schlüssel von Alice die Prüfsumme

```
openssl␣rsautl␣-verify␣-in␣digest.signed␣\
                -out␣message.digest1␣-pubin␣-inkey␣alice.public↵
```

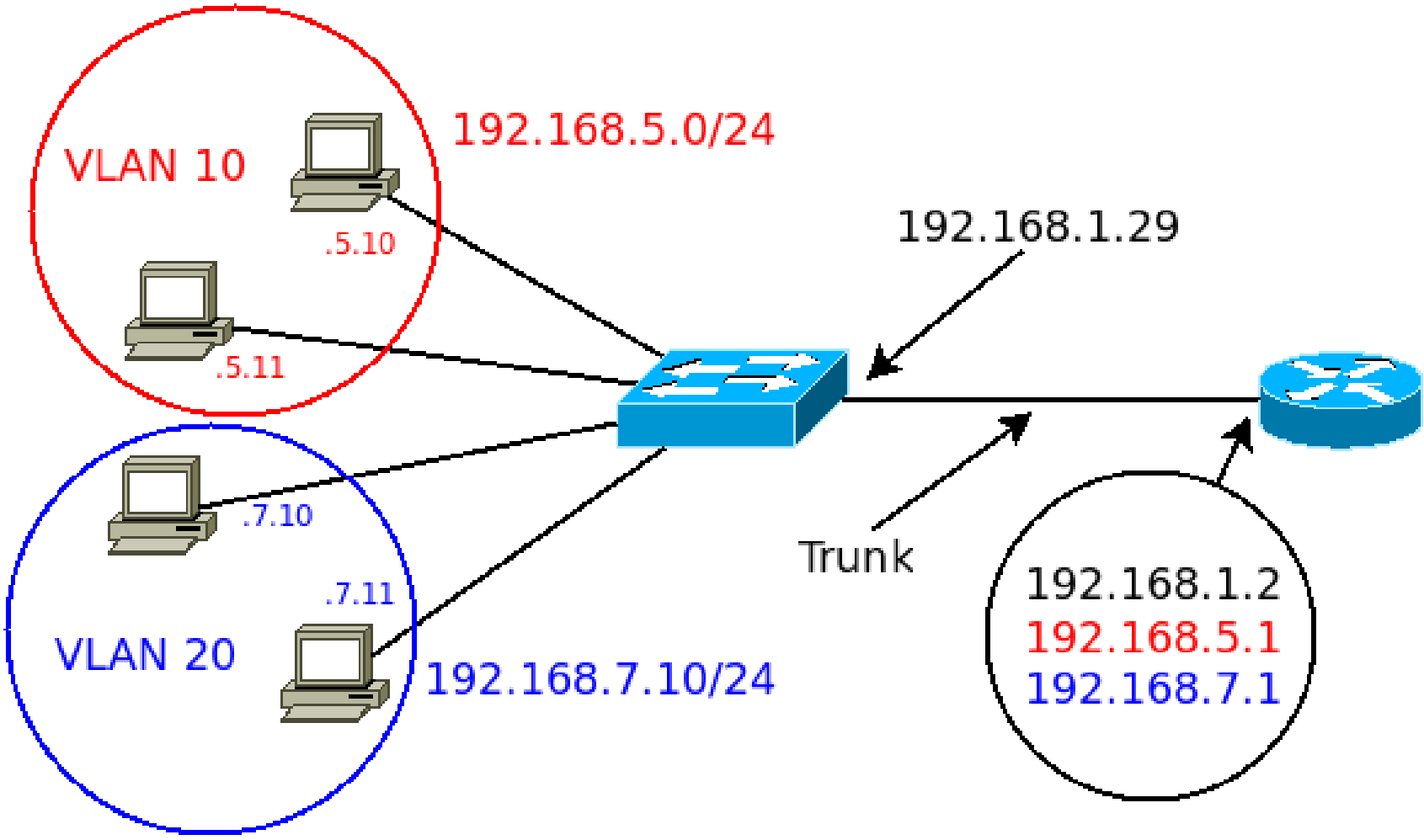
- Bob berechnet mit dem öffentlichen Schlüssel von Alice ebenfalls die Prüfsumme über der Nachricht

```
openssl␣dgst␣-binary␣message.plain␣>message.digest2↵
```

- Bob vergleicht die beiden Prüfsummen: Wenn sie identisch sind, handelt es sich um die unverfälschte Nachricht von Alice.

```
diff␣message.digest1␣message.digest2↵
```

Die folgende VLAN-Topologie soll in Betrieb genommen werden:



```
!-- Schritt 1: Zustand des Switches feststellen.
!-- Nach dem Booten sollte zuerst das NVRAM überprüft werden.
!-- Neue Switches enthalten nämlich keinerlei Konfigurations-
!-- dateien. Das bedeutet, alle Ports sind im VLAN 1 und der
!-- Switch kann sofort in Betrieb genommen werden.
!-- Mit folgendem Kommando wird der Inhalt des NVRAM's angezeigt:
!
Switch>enable↵
Switch#dir␣flash:↵
!
!
!-- Schritt 2: Benennen des Switches und
!-- Zugangskontrollen vereinbaren
!
Switch#conf␣term↵
Switch(config)#hostname␣C2950-24_NWL↵
C2950-24_NWL(config)line␣con␣0↵
C2950-24_NWL(config-line)password␣cisco↵
C2950-24_NWL(config-line)login↵
C2950-24_NWL(config-line)line␣vty␣0␣15↵
C2950-24_NWL(config-line)password␣cisco↵
C2950-24_NWL(config-line)login↵
C2950-24_NWL(config-line)exit↵
```

```
!-- Schritt 3: Absichern des Switches gegenüber
!-- unerlaubtem Befehlsmoduszugriff
!-- Das enable password dient der User Access Verification.
!-- D.h. nur wer dieses Passwort kennt, kann sich später,
!-- z.B. per telnet, mit dem Switch verbinden.
!-- Das enable secret Passwort wird benötigt, um in den
!-- enable Modus zu kommen.
!-- Das enable sectrat Passwort wird verschlüsselt gespeichert!!!
!
C2950-24_NWL(config)#enable␣password␣cisco↵
C2950-24_NWL(config)#enable␣secret␣class↵
C2950-24_NWL(config)#end↵
```

```
!-- Schritt 4: Konfigurieren des Switches
!-- mit einer IP-Adresse.
!-- WICHTIG: Die IP-Adresse wird nicht einer physischen
!-- Schnittstelle (also einem Ethernet-Port)
!-- explizit zugewiesen. Vielmehr bekommt das VLAN 1 (also das
!-- sogn. Management VLAN) diese Adresse zugewiesen.
!-- Damit wird sozusagen eine logische Schnittstelle (nämlich
!-- ein VLAN) konfiguriert.
!-- Es ist darauf zu achten, daß später immer ein Port in
!-- diesem Management VLAN verbleibt, weil der Switch sonst
!-- nur noch über den Console-Port zu programmieren ist, man sich
!-- sonst also selbst aussperren würde (und das ist sehr unangenehm,
!-- weil dann alle Wartungsarbeiten direkt am Switch durchgeführt
!-- werden müssen).
!-- Folgende Befehle weisen dem Switch eine IP-Adresse und
!-- eine Netzwerkmaske zu:
!
C2950-24_NWL#configure␣terminal↵
C2950-24_NWL(config-if)#interface␣vlan␣1↵
C2950-24_NWL(config-if)#ip␣address␣192.168.1.29␣255.255.255.0↵
C2950-24_NWL(config-if)#exit↵
```

```
!-- Schritt 5: Default Gateway festlegen:
!
C2950-24_NWL(config)#ip␣default-gateway␣192.168.1.1↵
C2950-24_NWL(config)#exit↵
!
!
!-- Schritt 6: VLAN 1 starten:
!
C2950-24_NWL#configure␣terminal↵
C2950-24_NWL(config-if)#interface␣vlan␣1↵
C2950-24_NWL(config-if)#no␣shutdown↵
C2950-24_NWL(config-if)#exit↵
C2950-24_NWL(config-if)#exit↵
!
!
!-- Schritt 7: Erzeugen und Benennen von 2 VLAN's mit
!-- den folgenden Befehlen:
!
C2950-24_NWL#vlan␣database↵
C2950-24_NWL(vlan)#vlan␣10␣name␣Verkauf↵
C2950-24_NWL(vlan)#vlan␣20␣name␣Support↵
C2950-24_NWL(vlan)#exit↵
```

```
!-- Schritt 8: Den VLANS werden jetzt
!-- Ethernet-Ports zugewiesen. Mit den folgenden
!-- Befehlen werden die Ports
!-- 0/5 bis 0/8 dem VLAN 10 hinzugefügt:
!
C2950-24_NWL#configure␣terminal↵
C2950-24_NWL(config)#interface␣fastethernet␣0/5↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣10↵
C2950-24_NWL(config-if)#interface␣fastethernet␣0/6↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣10↵
C2950-24_NWL(config-if)#interface␣fastethernet␣0/7↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣10↵
C2950-24_NWL(config-if)#interface␣fastethernet␣0/8↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣10↵
C2950-24_NWL(config-if)#end↵
```

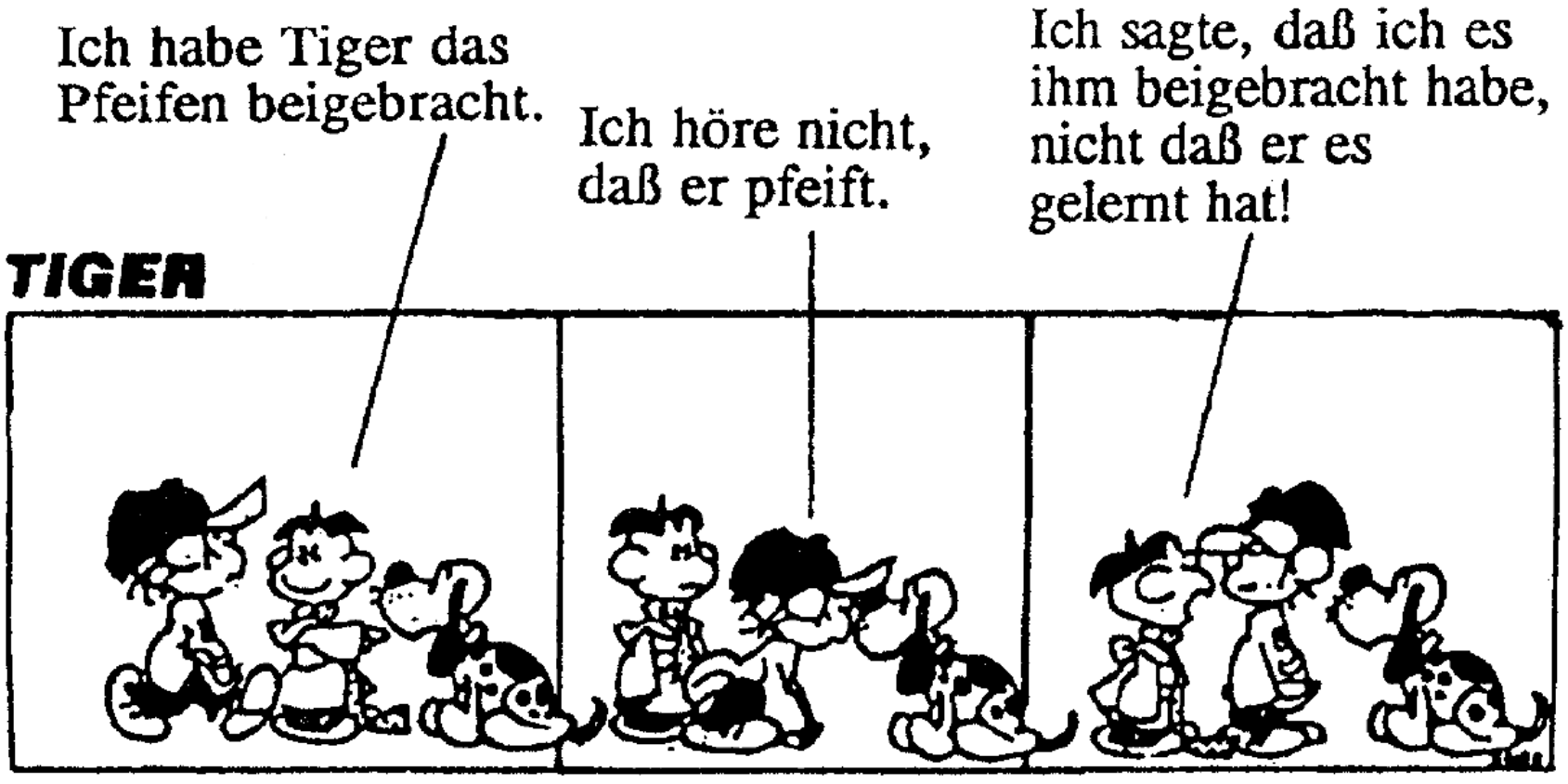
```
!-- Schritt 9: s.a. Schritt 8. Nur werden hier die
!-- Ethernet-Ports 0/9 bis 0/12
!-- dem VLAN 20 hinzugefügt:
!
C2950-24_NWL#configure␣terminal↵
C2950-24_NWL(config)#interface␣fastethernet␣0/9↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣20↵
C2950-24_NWL(config-if)#interface␣fastethernet␣0/10↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣20↵
C2950-24_NWL(config-if)#interface␣fastethernet␣0/11↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣20↵
C2950-24_NWL(config-if)#interface␣fastethernet0/12↵
C2950-24_NWL(config-if)#switchport␣mode␣access↵
C2950-24_NWL(config-if)#switchport␣access␣vlan␣20↵
C2950-24_NWL(config-if)#end↵
```

```
!-- Schritt 10: Überprüfen der VLAN-Konfiguration
!
C2950-24_NWL#show␣vlan↵
!
!
!-- Schritt 11: Trunking auf Port 0/1 (Catalyst 3524, bzw.
! 2/1 für Catalyst 2916XL) anschalten
!
C2950-24_NWL(config)#interface␣fastethernet␣0/1↵
C2950-24_NWL(config-if)#switchport␣mode␣trunk↵
!-- Die nächste Zeile nur für Catalyst 3524 bzw. Catalyst 2916XL
!-- C2950-24_NWL(config-if)#switchport␣trunk␣encapsulation␣dot1q↵
C2950-24_NWL(config-if)#switchport␣trunk␣allowed␣vlan␣1,10,20↵
C2950-24_NWL(config-if)#switchport␣trunk␣native␣vlan␣1↵
C2950-24_NWL(config-if)#end↵
```

```
!-- Schritt 12: Zunächst werden beim Router
!-- der Hostname (C2600XM-NWL) sowie die
!-- Console, VTY, und das enable password konfiguriert.
!-- Das enable secret password ist class, das normale Passwort cisco.
!
Router>show␣flash↵
Router>enable↵
Router#conf␣term↵
Router(config)#hostname␣C2600XM-NWL↵
C2600XM-NWL(config)#line␣con␣0↵
C2600XM-NWL(config-line)#password␣cisco↵
C2600XM-NWL(config-line)#login↵
C2600XM-NWL(config-line)#line␣vty␣0␣15↵
C2600XM-NWL(config-line)#password␣cisco↵
C2600XM-NWL(config-line)#login↵
C2600XM-NWL(config-line)#exit↵
C2600XM-NWL(config)#enable␣password␣cisco↵
C2600XM-NWL(config)#enable␣secret␣class↵
C2600XM-NWL(config)#end↵
```

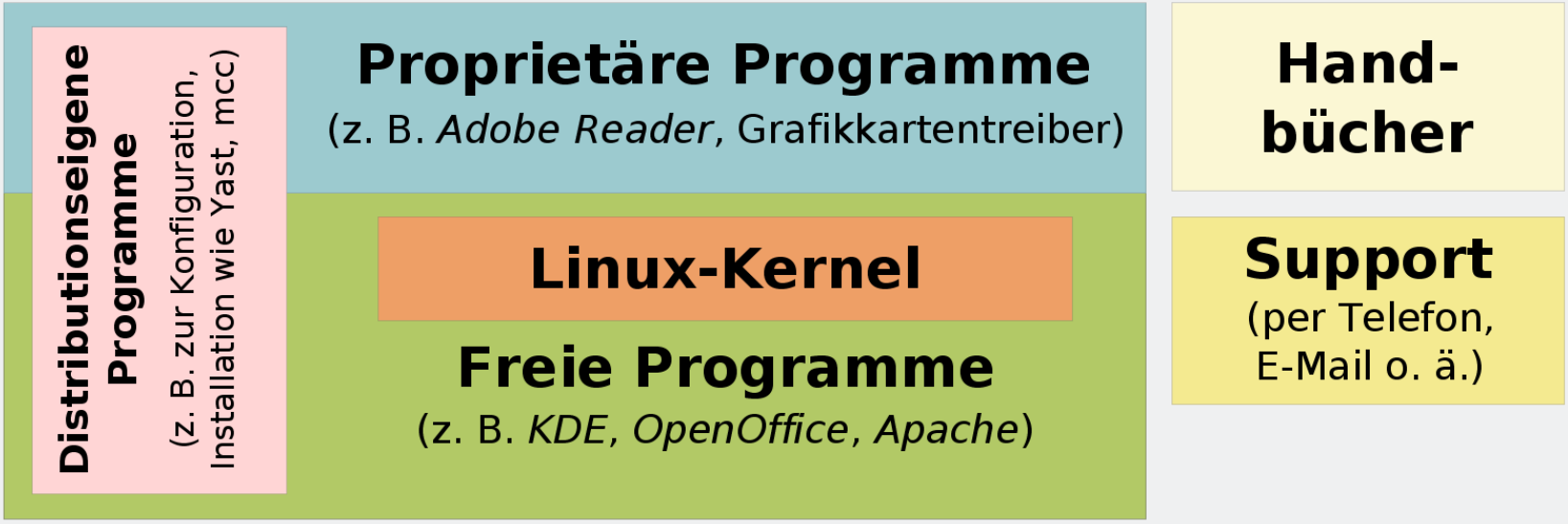
```
!-- Schritt 13: Interface-Konfiguration
!
C2600XM-NWL#conf␣term↵
C2600XM-NWL(config)#interface␣ethernet␣0/0↵
C2600XM-NWL(config-if)#no␣shutdown↵
C2600XM-NWL(config-if)#interface␣ethernet␣0/0.1↵
C2600XM-NWL(config-subif)#encapsulation␣dot1q␣1␣native↵
C2600XM-NWL(config-subif)#ip␣address␣192.168.1.2␣255.255.255.0↵
C2600XM-NWL(config-subif)#exit↵
C2600XM-NWL(config-if)#interface␣ethernet␣0/0.2↵
C2600XM-NWL(config-subif)#encapsulation␣dot1q␣10↵
C2600XM-NWL(config-subif)#ip␣address␣192.168.5.1␣255.255.255.0↵
C2600XM-NWL(config-subif)#exit↵
C2600XM-NWL(config-if)#interface␣ethernet␣0/0.3↵
C2600XM-NWL(config-subif)#encapsulation␣dot1q␣20↵
C2600XM-NWL(config-subif)#ip␣address␣192.168.7.1␣255.255.255.0↵
C2600XM-NWL(config-subif)#end↵
```

Alles klar?


Ich habe Tiger das Pfeifen beigebracht.
Ich höre nicht, daß er pfeift.
Ich sagte, daß ich es ihm beigebracht habe, nicht daß er es gelernt hat!
TIGER

Distribution

z. B. Debian, Red Hat, SUSE, Mandriva



zurück