

Bezeichnung	Speicherkap.	Erw.	Preis (pro GB)	Geschwindigkeit	Relevanz
Festplatte (FireWire)	500 GB	Nein	0,18 €	400 MB/s (?)	aktuell
HD Diskette (USB)	1,44 MB	Ja	423,00 €		eol
optisches LW (SATA)	bis zu 3,9 TB	Ja	0,10 €	200 MB/s	aktuell
SD-Karten	128 GB	Ja	1,00 €	30 MB/s	aktuell
Festplatte	2 TB	Nein	0,05 €	200 MB/s	aktuell
Memorystick	256 GB	Ja	1,60 €	60 MB/s	aktuell
Flashspeicher	128 GB	Nein	1,00 €	500 MB/s	aktuell
Magnetbänder (Streamer)	3 TB	Ja	0,02 €	160 MB/s	aktuell
FTP SDSL	unbegr.	Ja	0,01 – 0,70 €	0,7 MB/s	aktuell
RAID 0 (o. Red.)	2 x kleinste HD	Nein	0,05 €	?	aktuell
RAID 1 (voll Red.)	2 x HD gespiegelt	Nein	0,10 €	?	aktuell
RAID 5 (voll Red.) Hot Swap	(n-1) x kleinste Pl.	Nein	0,05 < x < 0,10 €	?	aktuell

Analyse einer Datensicherung

Preisfrage ist das wichtigste Kriterium!

1. Was soll gesichert werden:
 - Hardware gegen Ausfall
 - Hardware gegen Brand, Diebstahl usw.
 - Betriebssystem (Windows Server)
 - Datensicherung
2. Datenmengen
3. Intervall der Datensicherung (Zeitaspekt, Haltbarkeit)
 - Komplettsicherung (Schule)
 - Zeitdefinierte Teilsicherungen (Sekretariat, Bodden)
 - Synchronsicherung (Buchhaltung)
4. Ausfallzeit
 - ~24h: Schule und ähnliche Einrichtungen
 - Zeitfenster: 4-24 h: Kleine Firmen, die nicht unmittelbar vom PC abhängen (Galabau)
 - Zeitfenster: < 4h: Verwaltung, Uni, Hotels, Arztpraxis,...
 - Sekundengenau: Krankenhaus, Steuerberater, ...

Kleinbetrieb/Schule

Daten

- externe Sicherung auf NAS/HD (USB) (1,5TB)
- Sicherung mit einer speziellen Software (Acronis, xcopy in Batch-Datei)
- Hardwareabsicherung: Reserve Motherboard, Image der HD, Wandschrank
- Datensicherung außer Haus (2. ext. Festplatte)

Kosten <300 € + Wandschrank (+200,-)

4-24h

Daten

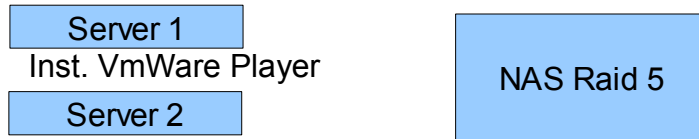
- Raid 1: Daten bleiben erhalten. System bleibt evtl. stehen, Lösung: Anfahrt und defekte Platte entfernen.
- Mgl 1: Rest wie oben (Motherboard, Image, Wandschrank)
- Mgl 2: ext. Sicherung über FTP (wg. Diebstahls)

Echtzeit

Betriebssystem:

Grundvoraussetzung USV (Strom für ca. 10 Minuten)

- Redundante Server (zweiter Server auf Standby)

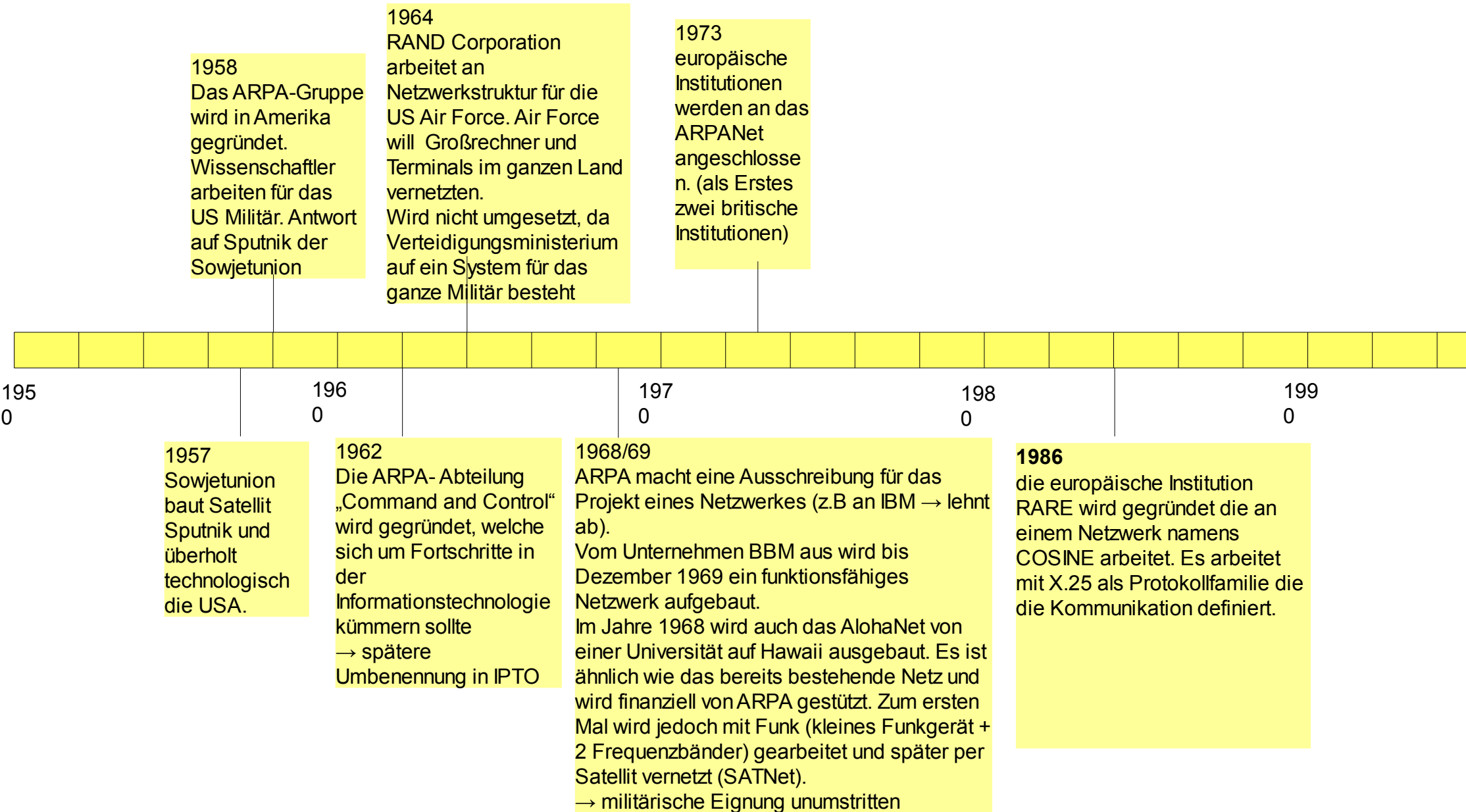


- Daten sind durch den NAS und Raid 5 gesichert. HotSwap (HD kann im laufenden Betrieb getauscht werden)
- Ist höhere Gewalt abzusichern, so sind externe Sicherung wie z.B. FTP oder Medien nötig
- Aktive Wandschränke (mit Kühlung)
- Zum Schutz vor versehentlichem Löschen, könnte eine geschützte Partition verwendet werden, auf der die User nur Leserechte (oder keine) haben. Ein Serverprozess kopiert additive alle Dateien auf diese Partition incl. Änderungen. Dazu gibt es spezielle Programme nach dem Prinzip von xcopy ...

Ergänzungen

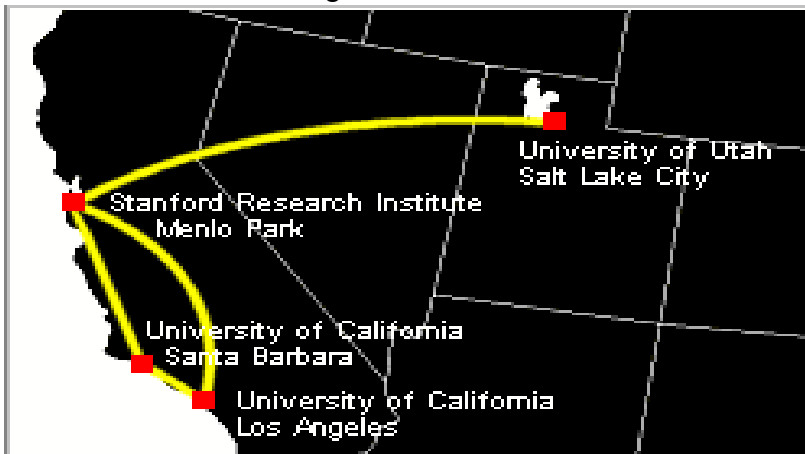
- Rechtliches: Es besteht die Verpflichtung zur Aufbewahrung (Finanzamt 10 J.) Verantwortung liegt zunächst beim Firmenchef. Beauftragte haften in ihrem Rahmen bis hin zur Entlassung.
- Gefährdungskatalog: Temperaturschwankung, Luftfeuchtigkeiten usw. Organisatorische Mängel bei der Datensicherung, Fehlbedienung, techn. Versagen, vorsätzliche Handlungen.
- Absicherung von versehentlichen Schäden durch Fremdpersonal.
- Ablaufplan im Falle eines Ausfalls

Netzwerktechnik (geschichtliches)

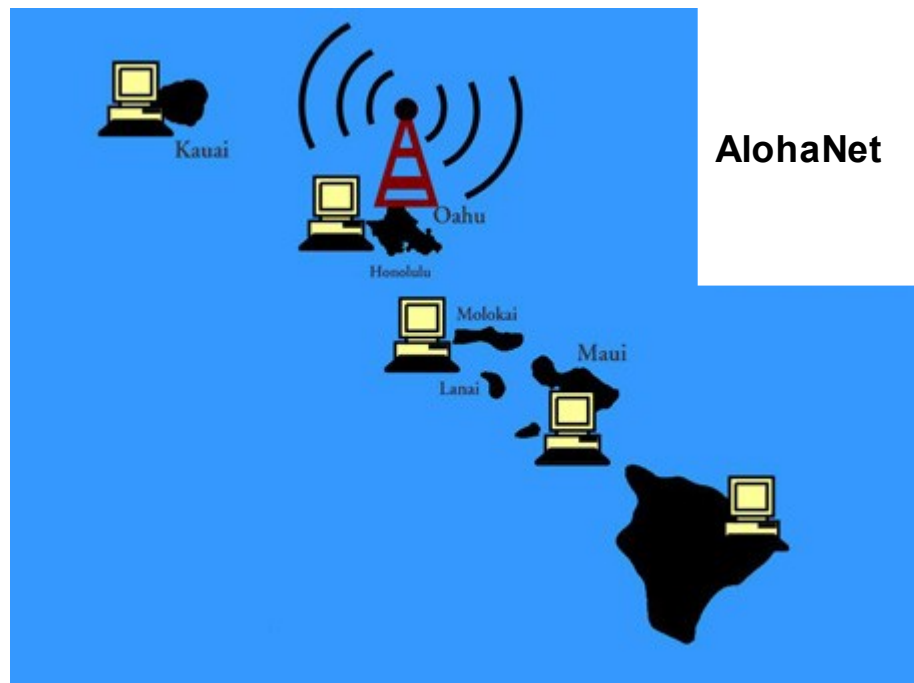


Das erste funktionsfähige Netzwerk zwischen vier Universitäten:

AlohaNet-



Schema:



Anforderungen:

- Verlustfrei arbeiten. Ein PC fällt aus, Rest kann weiter arbeiten.
- Datenverlustsicherheit. Die Daten werden in Portionen verpackt und alles muss ankommen.
- Abhörsicherheit

RFC

Request for Commons
Bitte um Kommentare

RFCs sind eine durchnummerierte Serie von Dokumenten, die verschiedene tatsächliche und vorgeschlagene Gewohnheiten beschreiben, die einen Bezug zum Internet haben und von der IETF (Internet Engineering Task Force) herausgegeben werden. Die Sammlung ist sowohl hinsichtlich des Themas, als auch des sogenannten Status, uneinheitlich. Protokolle sind für die Zusammenarbeit der Systeme unentbehrlich;

Programme, die untereinander Daten austauschen, müssen auf einigen Übereinstimmungen hinsichtlich des Datenformates und verwandten Themen beruhen. Sie sind also Dokumente, die der Öffentlichkeit zur Verfügung gestellt werden, damit diese darüber diskutiert.

Jeder RFC besitzt einen Status für die Gültigkeit. Hier einige Beispiele:

- Informational – Hinweis, Idee, Nutzung
- Experimental – zum Experimentieren
- Proposed Standard – Vorschlag für Standard
- Draft Standard – Begutachtung von mindestens zwei unabhängigen Implementierungen
- Standard – offizieller Standard STDn
- Historic – nicht mehr benutzt

Erfolg durch Formalismus

RFCs sind extrem formalistisch gestaltet:

- Vorschläge für neue oder geänderte RFCs werden in allen Änderungen vor der formellen Veröffentlichung nachvollziehbar dokumentiert.
- Ein einmal abschließend veröffentlichter RFC ist für immer öffentlich und fest. Er kann auch nicht korrigiert, sondern nur durch neuere RFCs abgelöst werden.
- Wie man RFCs schreibt, ist in RFC 2223 festgelegt.
- In RFC 2119 ist beschrieben, welche Bedeutung bestimmte Begriffe haben. Selbst Begriffe wie MUST oder MUST NOT werden in ihrer Bedeutung klar definiert, um Verwirrung in deren Interpretation zu vermeiden.
- Auch werden z. B. Strings und ihre Zusammensetzung formalistisch mittels Backus-Naur-Form (BNF) dargestellt. Dies sorgt für eine eindeutige Interpretation, hilfreich z. B. beim Aufbau von URLs/URIs.

All diese Formalismen sorgen für die Vermeidung von Missverständnissen in der Interpretation und Implementierung und somit für den Erfolg u. a. des Internets. Als Beispiele hierfür und für ihren Erfolg seien RFC 2822 (E-Mail) sowie RFC 2616 (HTTP) genannt.

Eine Liste alle RFCs können über die Seite ietf.org eingesehen werden. Dort werden alle Absprachen, die die Kommunikation in einem Netzwerk regelt, offengelegt.

Insbesondere der jüngste Adressengap wird durch ein RFC geregelt.

IPv4 war der bisherige Standard. Eine IP-Adresse besteht aus vier Blöcken mit jeweils 256-Zuständen:

255.255.255.255 Somit gibt es 2^{32} Zustände, also ca. 4,3 Mrd

Die Adressen wurde großzügig verschwendet. Bsp: 127.x.x.x Loopback Adapter

Jetzt IPv6 6 Blöcke zu je $2^8 = 256$ Zuständen 2^{128} Möglichkeiten.

OSI Schichtenmodell

Beschreibt den Ablauf vom Programm bis zum Kabel

- 1) Kabel (Mittel der Übertragung)
- 2) Sicherungsschicht (zuverl. Austausch)
- 3) Vermittlungsschicht (richtige Adresse)
- 4) Transportschicht (Aufz. der Daten in Pakete mit Fehlerkontrolle) TCP/UDP/FTP
- 5) Sitzungsschicht (Fehlerkorrektur)
- 6) Darstellungsschicht (Pakete werden wieder zusammengesetzt)
- 7) Anwendungsschicht (Schnittstelle zum Anwender)

Die versch. Protokolle setzen in den unterschiedlichen Schichten an.

Auflistung der meist benutzten Protokolle:

Transportschicht:	TCP/UDP/FTP/SCTP (Transportprot.)
Anwendungsschicht:	HTTP Darstellung von Webseiten
	SMTP Verwaltung von Geräten in einem Netzwerk.
	POP/IMAP Emailaustausch
	TELNET Remotezugriff
	SSH Remotezugriff
	IGMP Anmeldung am Router für Multicastanfr. Internet group Management prot.
	ICMP Internet Control Message Prot.
Vermittlungsschicht:	IP Datenpaketübertragung

Datenaustausch

- Cross-Over-Kabel ist notwendig (Lan-Kabel funktioniert nur mit Router)
- gleiche Arbeitsgruppe
- Netzwerkkarte
- Ordner müssen freigegeben sein

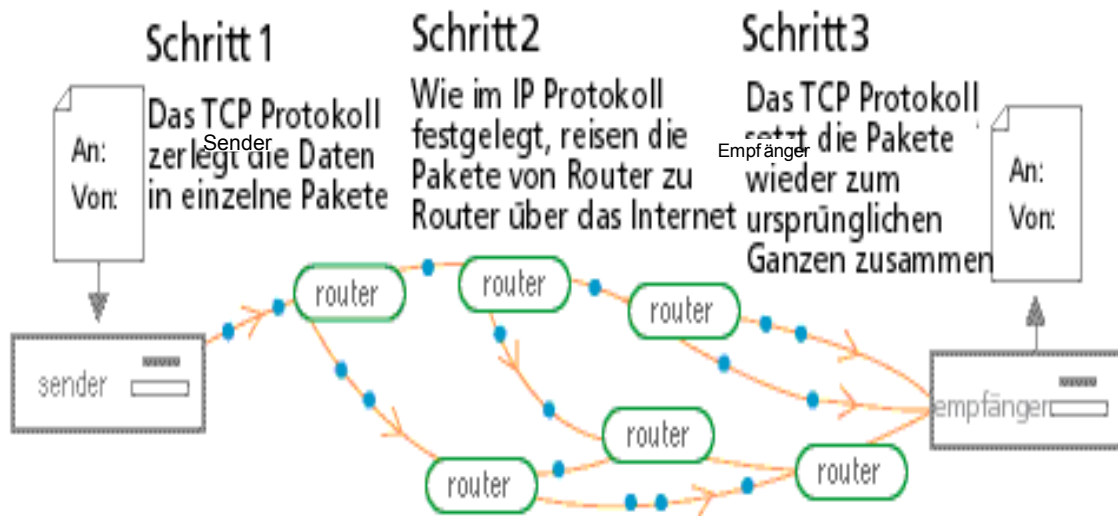
Transmission Control Protocol

1. Geschichte:
 - in 70er Jahre von USA entwickeltes und implementiertes Paket an Protokollen für World Area Networks
2. Funktionsweise:
 - Aufbau und Abbau von Verbindungen zwischen den Arbeitsstationen im Netzwerk
 - Steuert Datenfluss

Internet Protocol

1. Aufgabe:
 - Organisation und Adressierung der Daten
 - Zerlegt Daten → Datenpakete → Zusammensetzung am Ziel

Funktionsweise des Internets



Aufbau eines Datenaustauschs

Der eigentliche Datenaustausch und somit die wichtigste Identifikation ist die MAC-Adresse (Media Access Control)

Aufbau: ff:ff:ff:ff:ff:ff Zahlenbereich von 0-f also 16 Zustände

16^{12} ($(2^4)^{12} = 2^{48}$ Möglichkeiten)

Jedes Datenpaket enthält somit die MAC-Adresse des Senders und des Empfängers. Die IP-Adresse dient zur Gruppierung, der Rechnername der Bequemlichkeit.

Am Anfang schickt der Sender ein sog. Broadcast ins Netzwerk mit dem gesuchten Rechnernamen. Ex. ein DNS-Server, so wird dort die zugehörige IP-Adresse ausgegeben. Ex. kein DNS-Server dann nennt man das Netzwerk ein Peer-to-Peer Netzwerk. Der erste eingeschaltete Rechner übernimmt Serverfunktion und führt eine LMHOST-Tabelle. Dort werden alle Rechner eingefügt und die Namen mit IP-Adresse verteilt.

Das Protokoll ARP versucht nun die IP-Adresse in die MAC-Adresse aufzulösen. Dazu wird ein Datenpaket ins Netz geschickt, das als Absender die MAC-Adresse des Senders enthält und als Ziel die IP-Adresse des Empfängers. Der Empfänger ersetzt seine MAC-Adresse in Paket und schickt es zurück. Ex. der PC nicht im Netzwerk, so erzeugt das ICMP eine Fehlermeldung.

Konnte die MAC-Adresse ermittelt werden, so wird die Information in den ARP-Cache eingetragen.

Ist ein Eintrag im ARP-Cache vorhanden, so wird dieser grundsätzlich genutzt.

Mit arp -a kann die Tabelle eingesehen werden

Mit arp -s können Einträge von Hand eingetragen werden, mit -d gelöscht werden.

Dynamisch ermittelte Einträge werden nach ca. 20 Sekunden gelöscht.

ARP arbeitet nur im eigenen Netzwerk. Soll ein PC in einem anderen Netzwerk erreicht werden, so muss die Anfrage geroutet werden.

Wenn die IP-Adresse oder der Name nicht im eigenen Netzwerk ist, so wird die Anfrage an die Gateway-Adresse weitergegeben. Dort befindet sich ein Router, der die IP-Adresse des Datenpaketes austauscht. Dazu ersetzt er die Sender-Mac-Adresse gegen seine eigene MAC-Adresse.

Aufbau der IP-Adressen:

Es gibt drei Netzklassen

CLASS A-C

Bsp: 192.168.178.1 im PC werden diese Zahlen binär geführt:

255 .255 .255 .255
11111111.11111111.11111111.11111111

$$2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0$$

$$1 0 1 0 0 0 0 1 = 2^7 + 2^5 + 2^0 = 128 + 32 + 1 = 161$$

Die Subnetmaske entscheidet, welche PCs in einem Netz sind.

IP-Adresse: 11000000.00000001.10000111.11101100

Subnetmaske: 11111111.11111111.11111111.00000000

UND: 11000000.00000001.10000111.00000000

Netzwerkteil . Host

Class A: >00000000.x.x.x bis < 10000000.x.x.x

Class B: >10000000.x.x.x bis < 11000000.x.x.x

Rest ist Class C

Aufgabe: Zwei PCs sollen sich gegenseitig nicht sehen, aber beide auf einen Server zugreifen, und somit ins Internet gehen können.

Lösung im Class A

```
WS1 10.10.1.10:  00001010.00001010.00000001.00001010
SM 255.255.255.0 11111111.11111111.11111111.00000000
UND              00001010.00001010.00000001.00000000
```

```
WS5 10.10.2.10:  00001010.00001010.00000010.00001010
SM 255.255.255.0 11111111.11111111.11111111.00000000
UND              00001010.00001010.00000010.00000000
```

```
SVR:10.10.3.1:   00001010.00001010.00000011.00000001
SM 255.255.248.0 11111111.11111111.11111000.00000000
UND              00001010.00001010.00000000.00000000
```

ROUTER

```
10.10.4.1        00001010.00001010.00000100.00000001
SM 255.255.248.0 11111111.11111111.11111000.00000000
UND              00001010.00001010.00000000.00000000
```

DNS-Server

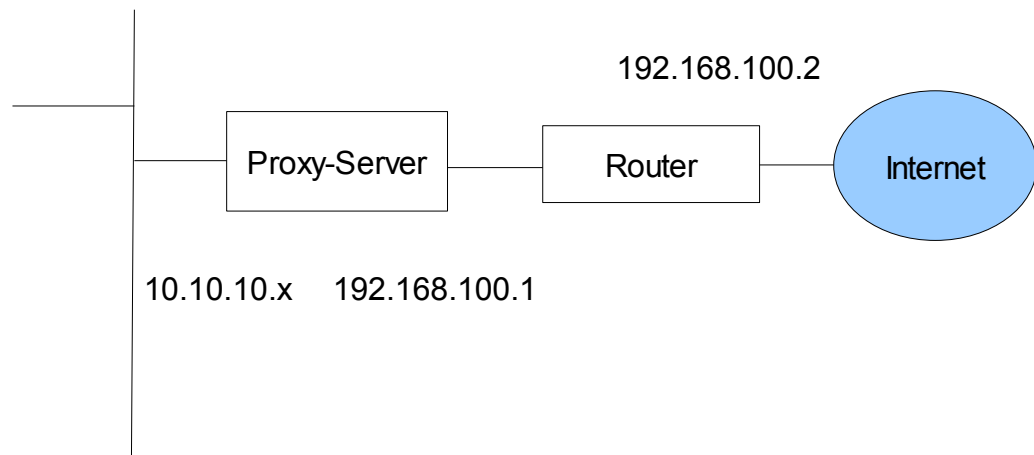
Aufgabe: Umwandlung des Domainnamens in die IP-Adresse.

Der Domänennamensraum

Root-Domäne	Top-Level-D.	Sub-D.	Server
„.de“	„.brother“	„.vertrieb“	„.www“ „.ww2“
„.“	„.com“	„.einkauf“	
	„.tc-aachen“		

Bei der Auflösung gibt es eine Forward-Lookupzone und eine Reverse-L.

Der Proxy-Server:



Proxy-Server wird zwischen Hausnetz und dem Router installiert. Die IP-Kreise sind sinnvollerweise völlig getrennt. Der Proxy-Server sollte ein nicht anfälliges Betriebssystem haben. Z. B. Linux.

Aufgaben:

- schnellerer Zugriff durch Caching (zwischenspeichern von Seiten)
- IP-Translation (Hausnetzadressen werden im Proxy ersetzt)
- Virenfiler bzw. DDOS Angriffe abfangen. Firewall
- Email-Server mit Spamfilter (Microsoft: Exchange)
- Portsperr (teilweise auch mit Router mgl.)
- Detaillierte Zugriffsregeln für IP-Adressen, Seiten, User

Aufbau eine heterogenen Netzwerkes mit Linux als PDC

- DHCP
- DNS-Server
- SAMBA
- AD bzw. Gegenstück bei Linux

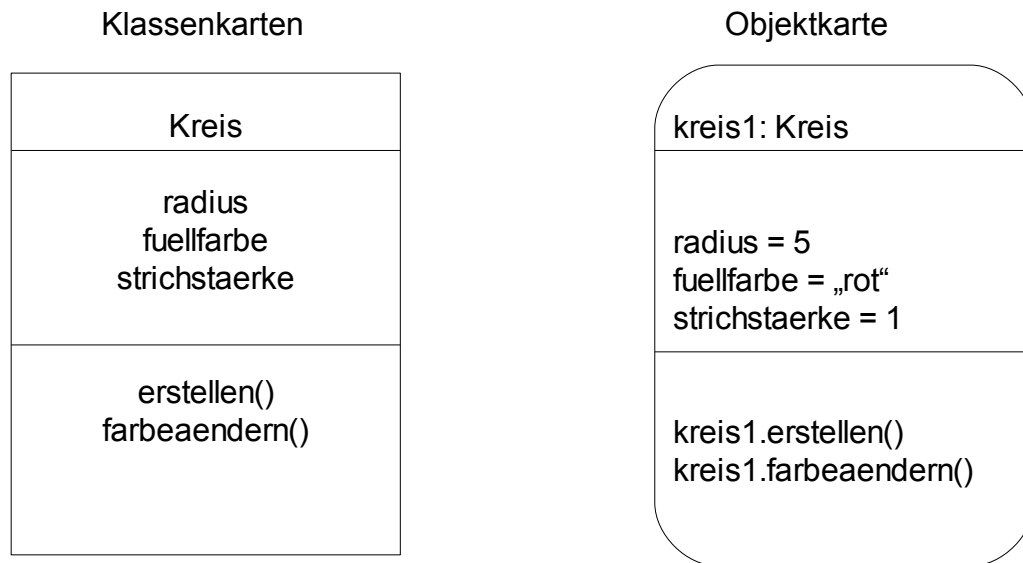
Testen der Funktionalität

- Rechner ohne feste IP im Netz muss die IP-Einstellungen erhalten
- nslookup auf einem WindowsClient sollte den Namen auflösen
- Netzwerklaufwerk verbinden, das auf dem Linuxserver liegt (Zugriffsrechte)
- PDC Primary-Domain-Controller
 - Zentrale Verwaltung der Anmeldeaccounts
 - Wiederherstellen des Profils (Usereigene Desktop)
 - Homelaufwerk

Netzwerk 1: Class C Netz: 192.168.100.x

Netzwerk 2: Class C Netz: 192.168.200.x

Grundlagen

**Algorithmus**

Möglichst genaue Beschreibung eines Handlungsablaufs oder einer Handlungsabfolge

Zustand

Der Zustand eines Objektes wird durch seine Attributwerte beschrieben. Ändert eine Methode ein Attribut, so hat sich der Zustand des Objektes verändert.

Die verschiedenen Zustände und deren Übergänge können in einem Zustandsdiagramm angegeben werden. Pfeile zwischen den Zuständen stellen die möglichen Übergänge dar.

Alle Vorgänge, die durch ein Zustandsdiagramm dargestellt werden können, heißen Zustandsautomaten.

Als Vorstufe zur Programmierung werden Algorithmen häufig in Struktogrammen dargestellt (ähnlich einem Ablaufdiagramm). Diese wurden von Nassi und Shneidermann definiert (DIN 66261 von 1985).

Zustandsdiagramme, die neben den verschiedenen Zuständen auch die auslösenden Aktionen beinhalten heißen endliche Automaten. Um alle möglichen Automaten erstellen zu können benötigt man noch bedingte Auslösehandlungen.

Diese werden durch Mehrfachverzweigungen implementiert.

Einfachverzweigung

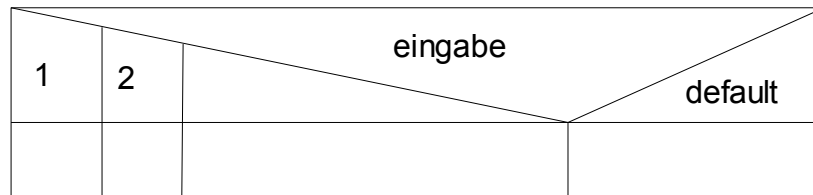
```
if (lBed) { }
else { }
```

Mehrfachverzweigungen

```

switch (eingabe) {
    case 1: { anweisung } break;
    case 2: { anweisung } break;
    ...
    default: { anweisung};
}

```



Typische Anwendungsbeispiele

1) Verschachtelte Einfachverzweigung

Kennworteingabe (Bsp: „wert“) beim Tresor

```

if (cEingabe.substr(0,1) == „w“ {
    if (cEingabe.substr(1,2) == „e“ {
        ...
    }
    else { AusgabeFehler() }
}
else { AusgabeFehler() }

```

2) Mehrfachverzweigungen mit switch

Fahrstuhlanforderung beim UG EG OG

```
public void zustandwechseln(cEingabe)
```

```

...
switch (cEingabe) {
    case „EG“: {
        if (status==„UG“) {hoch(1); status = „EG“}
        else if (status ==„OG“) {runter(1); status = „EG“}
        else { warte() } } break;
    case „OG“: {
        if (status==„UG“) {hoch(2); status = „OG“}
        else if (status ==„EG“) {hoch(1); status = „OG“}
        else { warte() } } break;
    case „UG“: {
        if (status==„OG“) {runter(2); status = „UG“}
        else if (status ==„EG“) {runter(1); status = „UG“}
        else { warte() } } break;
}

```

Formale Sprachen

Eingabealphabet: Alle möglichen Eingaben, die im Automaten berücksichtigt werden müssen. $\Sigma = \{ w, c, r \}$ für das Mischen eines Wodka-Cola Getränks mit rühren.

Formale Regeln beschreiben die Zulässigkeit beim Bilden von Wörtern (formale Sprache)

Die Gesamtheit aller Regeln beschreibt die Syntax der formalen Sprache, die Semantik ihre Bedeutung

Die Bildung eines Wortes aus den Regeln heißt auch „Ableitung“ des Wortes.

Jede Ableitung muss ausgehend von einer syntaktischen Variablen (S) beginnen. Diese syntaktischen Variablen müssen Schritt für Schritt in das Eingabealphabet aufgelöst werden.

Satzbildung in der deutschen Sprache:

$\Sigma = \{ a, \dots, z, \ddot{a}, \dots, \text{„“}, \text{““}, \dots, \text{„} \}$

\rightarrow <Hauptsatz>	\rightarrow <Subjekt> „ „ <Prädikat> [„ „ <Objekt>] „ „	R1
<Subjekt>	\rightarrow <Artikel> „ „ <Nomen>	R2
<Artikel>	\rightarrow <Buchstabe> [<Buchstabe>]	R3
<Nomen>	\rightarrow <Buchstabe> [<Buchstabe>]	R4
<Prädikat>	\rightarrow <Buchstabe> [<Buchstabe>]	R5
<Buchstabe>	$\rightarrow \Sigma$	