Hacking Workshop – Mathecamp 2016 in Windischleuba

Sven Prüfer

August 3, 2016

Hinweise

Rechtliches

Macht niemals irgendsoetwas auf Rechnern, auf denen ihr das nicht dürft oder von deren Betreibern ihr kein Einverständnis habt.

Rechtliches

Macht niemals irgendsoetwas auf Rechnern, auf denen ihr das nicht dürft oder von deren Betreibern ihr kein Einverständnis habt.

Und auf gar keinen Fall in der Schule!

Praktisches

Viele Menschen wollen euch Böses!

Praktisches

Viele Menschen wollen euch Böses!

Traut keinen zwielichten Websites, installiert niemals (besonders unter Windows) merkwürdige Programme!

Praktisches

Viele Menschen wollen euch Böses!

Traut keinen zwielichten Websites, installiert niemals (besonders unter Windows) merkwürdige Programme!

Informiert euch unbedingt über Skripte und Programme, bevor ihr sie ausführt!

Praktisches

Viele Menschen wollen euch Böses!

Traut keinen zwielichten Websites, installiert niemals (besonders unter Windows) merkwürdige Programme!

Informiert euch unbedingt über Skripte und Programme, bevor ihr sie ausführt!

Vertrauenswürdige Websites sind insbesondere stackoverflow.com, superuser.com oder news.ycombinator.com.

Linux – System

Dateistruktur

"Alles ist eine Datei" – Grundprinzip von Unix

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien
/home	Private Dateien der Nutzer

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien
/home	Private Dateien der Nutzer
/etc	Konfigurationsdateien, insb. /etc/ssl

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien
/home	Private Dateien der Nutzer
/etc	Konfigurationsdateien, insb. /etc/ssl
/var	Variable Dateien, insb. /var/www

Dateistruktur

“Alles ist eine Datei” – Grundprinzip von Unix

Das Wurzelverzeichnis ist “/” anstelle einer Partition (“C” unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien
/home	Private Dateien der Nutzer
/etc	Konfigurationsdateien, insb. /etc/ssl
/var	Variable Dateien, insb. /var/www
/bin	Binäre Dateien

Dateistruktur

"Alles ist eine Datei" – Grundprinzip von Unix

Das Wurzelverzeichnis ist "/" anstelle einer Partition ("C" unter Windows).

Wichige Verzeichnisse sind insbesondere:

/dev	Geräte
/media	Medien
/home	Private Dateien der Nutzer
/etc	Konfigurationsdateien, insb. /etc/ssl
/var	Variable Dateien, insb. /var/www
/bin	Binäre Dateien
/tmp	Temporäre Dateien

Benutzerrechte

Dateisystem speichert Lese-/Schreib-/Nutzungsrechte für jede einzelne Datei und jeden Ordner

Benutzerrechte

Dateisystem speichert Lese-/Schreib-/Nutzungsrechte für jede einzelne Datei und jeden Ordner

Bedeutung von Rechten bei Verzeichnissen anders.

Benutzerrechte

Dateisystem speichert Lese-/Schreib-/Nutzungsrechte für jede einzelne Datei und jeden Ordner

Bedeutung von Rechten bei Verzeichnissen anders.

Bei guter Nutzung von Rechten kann Eindringling im besten Fall nichts machen.

Benutzerrechte

Dateisystem speichert Lese-/Schreib-/Nutzungsrechte für jede einzelne Datei und jeden Ordner

Bedeutung von Rechten bei Verzeichnissen anders.

Bei guter Nutzung von Rechten kann Eindringling im besten Fall nichts machen.

Wichtigster Nutzer: *root*

Benutzerrechte

Dateisystem speichert Lese-/Schreib-/Nutzungsrechte für jede einzelne Datei und jeden Ordner

Bedeutung von Rechten bei Verzeichnissen anders.

Bei guter Nutzung von Rechten kann Eindringling im besten Fall nichts machen.

Wichtigster Nutzer: *root*

Beispiel in Konsole.

Die Kommandozeile

Terminal, Bash und Shell

Eine *Shell* verarbeitet Kommandozeilenbefehle und gibt eine Antwort.

Terminal, Bash und Shell

Eine *Shell* verarbeitet Kommandozeilenbefehle und gibt eine Antwort.

Die *Bash* ist die bekannteste Shell. Es gibt noch viele andere.

Terminal, Bash und Shell

Eine *Shell* verarbeitet Kommandozeilenbefehle und gibt eine Antwort.

Die *Bash* ist die bekannteste Shell. Es gibt noch viele andere.

Ein *Terminal* ist eine Art Verpackung für eine Shell, also z.B. das Fenster in dem die Shell läuft.

Wichtigste Befehle

cd	Wechsle Verzeichnis
ls	Zeige Verzeichnisinhalt
cat	Zeige/Gib wieder Inhalt von Textdateien an
man	Zeige Hilfe zu Befehl an
python/perl/gcc	Kompiliere mit entsprechender Sprache
sh	Führe Shellskript aus
DATEI	Führe binäre DATEI aus
make	Führe make Skript aus

Pipes

Befehle in der Bash können hintereinander ausgeführt werden mittels einer Pipe “|”. Diese gibt die Ausgabe als Eingabe an den nächsten Befehl weiter.

Pipes

Befehle in der Bash können hintereinander ausgeführt werden mittels einer Pipe “|”. Diese gibt die Ausgabe als Eingabe an den nächsten Befehl weiter.

```
cat testdatei | uniq -u | sort
```

Pipes

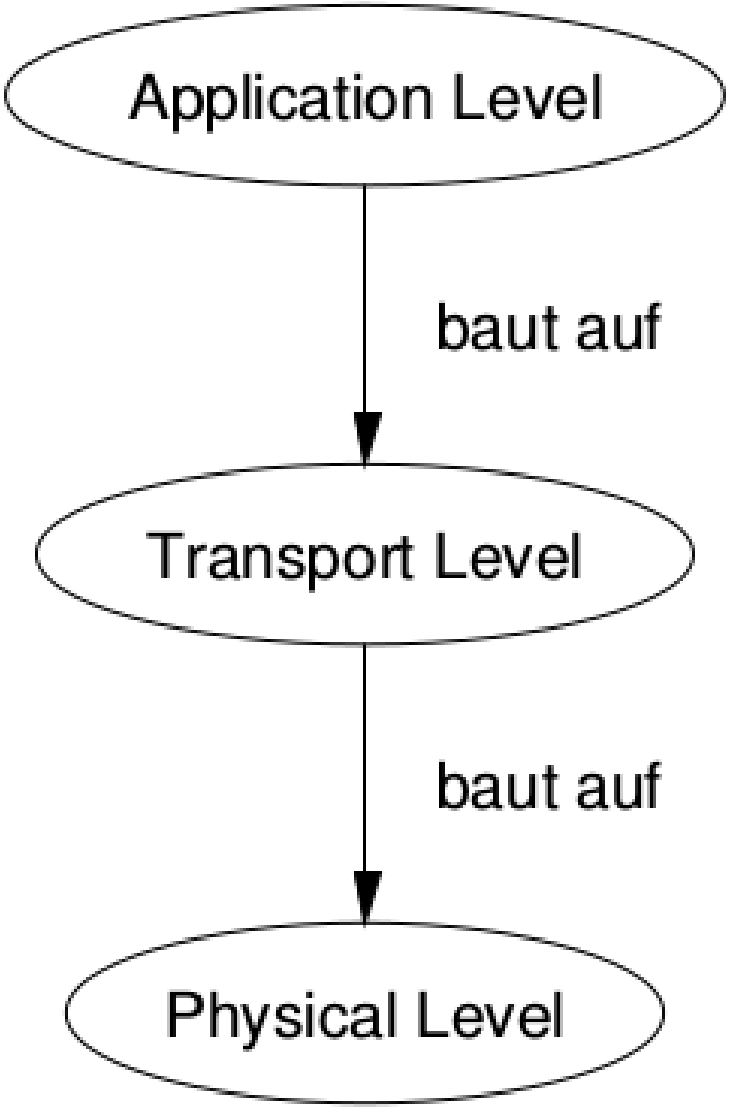
Befehle in der Bash können hintereinander ausgeführt werden mittels einer Pipe “|”. Diese gibt die Ausgabe als Eingabe an den nächsten Befehl weiter.

```
cat testdatei | uniq -u | sort
```

Gibt den Inhalt der Datei “testdatei” weiter an “uniq” mit Option “-u”, doppelte Zeilen werden weggeschmissen und danach sortiert.

Grundlagen Netzwerkkommunikation

Schichtenmodell



RFC

RFC = Request for Comments

RFC

RFC = Request for Comments

Internetstandards werden damit (in einfacher Textdatei) vorgeschlagen und zur Diskussion gestellt.

RFC

RFC = Request for Comments

Internetstandards werden damit (in einfacher Textdatei) vorgeschlagen und zur Diskussion gestellt.

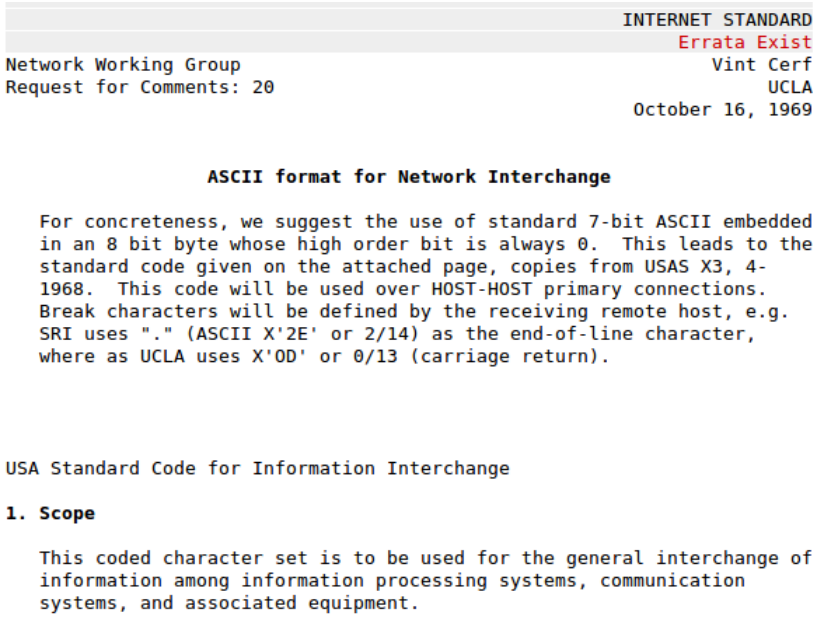
De facto werden Internetstandards damit definiert.

RFC

RFC = Request for Comments

Internetstandards werden damit (in einfacher Textdatei) vorgeschlagen und zur Diskussion gestellt.

De facto werden Internetstandards damit definiert.



INTERNET STANDARD
Errata Exist

Network Working Group — Vint Cerf
Request for Comments: 20 — UCLA
October 16, 1969

ASCII format for Network Interchange

For concreteness, we suggest the use of standard 7-bit ASCII embedded in an 8 bit byte whose high order bit is always 0. This leads to the standard code given on the attached page, copies from USAS X3, 4-1968. This code will be used over HOST-HOST primary connections. Break characters will be defined by the receiving remote host, e.g. SRI uses "." (ASCII X'2E' or 2/14) as the end-of-line character, where as UCLA uses X'OD' or 0/13 (carriage return).

USA Standard Code for Information Interchange

1. Scope

This coded character set is to be used for the general interchange of information among information processing systems, communication systems, and associated equipment.

IPs und DNS

IP-Protokoll

Grundlegendes Protokoll um Pakete vom Quell-Host zum Ziel-Host zu senden.

IP-Protokoll

Grundlegendes Protokoll um Pakete vom Quell-Host zum Ziel-Host zu senden.

Pakete bestehen aus “Header” und “Payload”.

IP-Protokoll

Grundlegendes Protokoll um Pakete vom Quell-Host zum Ziel-Host zu senden.

Pakete bestehen aus “Header” und “Payload”.

Es existieren zwei wichtige Versionen: IPv4 und IPv6.

IP-Protokoll

Grundlegendes Protokoll um Pakete vom Quell-Host zum Ziel-Host zu senden.

Pakete bestehen aus “Header” und “Payload”.

Es existieren zwei wichtige Versionen: IPv4 und IPv6.

Beispiel:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

Analyse etwas später!

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

10 – Type of Service (?)

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

10 – Type of Service (?)

00 2C – Länge des Pakets $\Rightarrow$ 44 Bytes

Beispiel IPv4 Paket 1

Beispiel (jedes "Paar" sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

10 – Type of Service (?)

00 2C – Länge des Pakets $\Rightarrow$ 44 Bytes

24 B2 – Durchnummerierung der Pakete

Beispiel IPv4 Paket 1

Beispiel (jedes “Paar” sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

10 – Type of Service (?)

00 2C – Länge des Pakets $\Rightarrow$ 44 Bytes

24 B2 – Durchnummerierung der Pakete

00 00 (Verschiedene IP Flags)

Beispiel IPv4 Paket 1

Beispiel (jedes "Paar" sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern $\Rightarrow$ 20 Bytes

10 – Type of Service (?)

00 2C – Länge des Pakets $\Rightarrow$ 44 Bytes

24 B2 – Durchnummerierung der Pakete

00 00 (Verschiedene IP Flags)

40 – "Time to live"

Beispiel IPv4 Paket 1

Beispiel (jedes "Paar" sind zwei Hexadezimalzahlen, also jeweils vier Bit, also insgesamt 1 Byte) von Michael Egan:

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

4 – IPv4

5 – Länge des IP Headers in 32-Bit-Wörtern ⇒ 20 Bytes

10 – Type of Service (?)

00 2C – Länge des Pakets ⇒ 44 Bytes

24 B2 – Durchnummerierung der Pakete

00 00 (Verschiedene IP Flags)

40 – "Time to live"

06 – Protokoll ⇒ TCP

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse $\Rightarrow$ 172.16.0.1

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse $\Rightarrow$ 172.16.0.1

Beginn TCP-Header: 04 47 – Quellport $\Rightarrow$ 1095

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse $\Rightarrow$ 172.16.0.1

Beginn TCP-Header: 04 47 – Quellport $\Rightarrow$ 1095

00 17 – Zielport $\Rightarrow$ 23 (Telnet)

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse $\Rightarrow$ 172.16.0.1

Beginn TCP-Header: 04 47 – Quellport $\Rightarrow$ 1095

00 17 – Zielport $\Rightarrow$ 23 (Telnet)

60 C6 DF 90 – Sequenznummer SEQ#

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse $\Rightarrow$ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse $\Rightarrow$ 172.16.0.1

Beginn TCP-Header: 04 47 – Quellport $\Rightarrow$ 1095

00 17 – Zielport $\Rightarrow$ 23 (Telnet)

60 C6 DF 90 – Sequenznummer SEQ#

00 00 00 00 – Bestätigungsnummer ACK# (normalerweise SEQ# des vorherigen Pakets, hier aber erstes Paket)

Beispiel IPv4 Paket 2

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

FD DF – Checksum des IP-Headers

AC 10 00 09 – Quell-IP-Adresse ⇒ 172.16.0.9

AC 10 00 01 – Ziel-IP-Adresse ⇒ 172.16.0.1

Beginn TCP-Header: 04 47 – Quellport ⇒ 1095

00 17 – Zielport ⇒ 23 (Telnet)

60 C6 DF 90 – Sequenznummer SEQ#

00 00 00 00 – Bestätigungsnummer ACK# (normalerweise SEQ# des vorherigen Pakets, hier aber erstes Paket)

6 – Länge des TCP Headers in 32-Bit-Wörtern ⇒ 24 Bytes

Beispiel IPv4 Paket 3

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10
00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00
F9 46 00 00 02 04 05 B4

0 02 – TCP Flags (?)

Beispiel IPv4 Paket 3

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

0 02 – TCP Flags (?)

02 00 – Fenstergröße für sogenanntes “Sliding Window Protocoll” zum Verhindern vom Senden zuvieler Pakete

Beispiel IPv4 Paket 3

45 10 00 2C 24 B2 00 00 40 06 FD DF AC 10 00 09 AC 10 00 01 04 47 00 17 60 C6 DF 90 00 00 00 00 60 02 02 00 F9 46 00 00 02 04 05 B4

0 02 – TCP Flags (?)

02 00 – Fenstergröße für sogenanntes “Sliding Window Protocoll” zum Verhindern vom Senden zuvieler Pakete

F9 46 – Checksum TCP-Header

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen:

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8,

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8, 172.16.0/12,

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8, 172.16.0/12, 192.168.0/16

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8, 172.16.0/12, 192.168.0/16

IPv4-Adressen sind alle vergeben :-(

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8, 172.16.0/12, 192.168.0/16

IPv4-Adressen sind alle vergeben :-(

Aber es gibt IPv6! :-)

IP-Adressen

IPv4 nutzt 4 Bytes um Rechner zu adressieren, z.B. 5.189.172.46.

Notation: 192.0.2.0/24 bezeichnet alle Adressen 192.0.2.0 bis 192.0.2.255

Spezielle Adressen: 10.0.0.0/8, 172.16.0/12, 192.168.0/16

IPv4-Adressen sind alle vergeben :-(

Aber es gibt IPv6! :-)

IPv6 nutzt 16 Bytes, typischerweise in Hexadezimal und ohne Nullen, z.B. 2001:0DB8:AC10:FE01::::

Routing

Rechner am Internet schicken nicht jedes Paket direkt an ihr Ziel

Routing

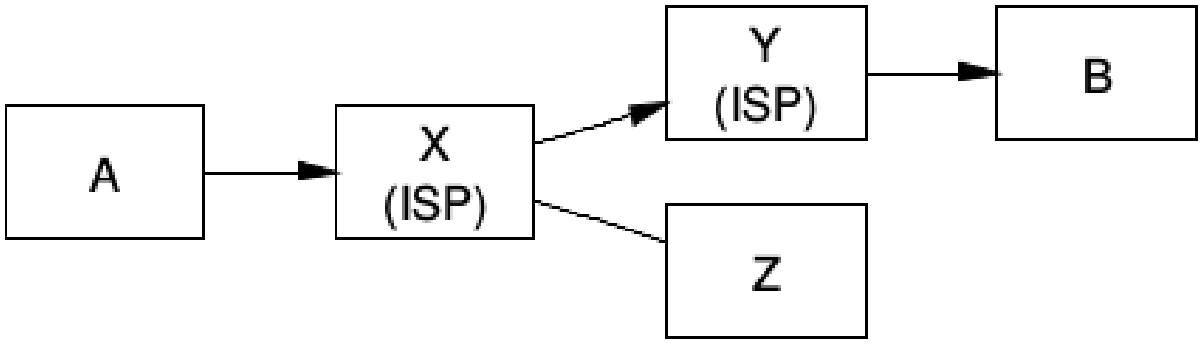
Rechner am Internet schicken nicht jedes Paket direkt an ihr Ziel

Stattdessen hat jeder Rechner eine *Routing-Tabelle* und schickt Pakete nach einem Routingprotokoll.

Routing

Rechner am Internet schicken nicht jedes Paket direkt an ihr Ziel

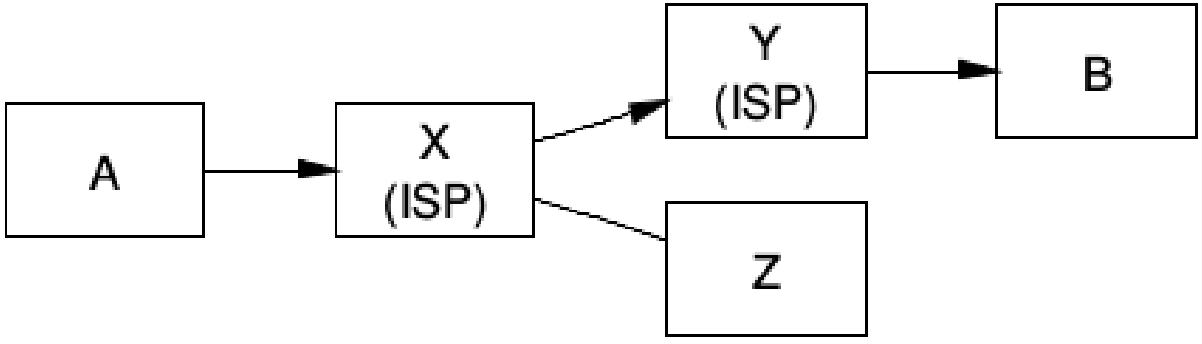
Stattdessen hat jeder Rechner eine *Routing-Tabelle* und schickt Pakete nach einem Routingprotokoll.



Routing

Rechner am Internet schicken nicht jedes Paket direkt an ihr Ziel

Stattdessen hat jeder Rechner eine *Routing-Tabelle* und schickt Pakete nach einem Routingprotokoll.

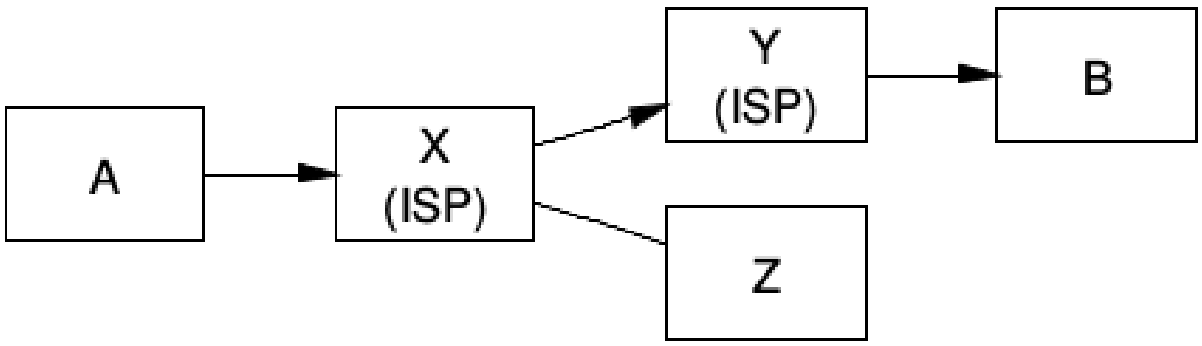


Vor 1993: IPs hierarchisch nach Größe der Netzwerke aufgeteilt $\Rightarrow$ Riesige Routingtables und nicht genügend IPs für LANs

Routing

Rechner am Internet schicken nicht jedes Paket direkt an ihr Ziel

Stattdessen hat jeder Rechner eine *Routing-Tabelle* und schickt Pakete nach einem Routingprotokoll.



Vor 1993: IPs hierarchisch nach Größe der Netzwerke aufgeteilt ⇒ Riesige Routingtables und nicht genügend IPs für LANs

Nach 1993: CIDR ⇒ Netzwerk- und Hostanteil der IP.

TCP 1

TCP = Transmission Control Protocol

TCP 1

TCP = Transmission Control Protocol

Erweiterung von IP um

TCP 1

TCP = Transmission Control Protocol

Erweiterung von IP um

- Fehlerkorrektur (Reihenfolge und Wiederholung) durch Nummerierung der Pakete

TCP 1

TCP = Transmission Control Protocol

Erweiterung von IP um

- Fehlerkorrektur (Reihenfolge und Wiederholung) durch Nummerierung der Pakete
- Ports

TCP 1

TCP = Transmission Control Protocol

Erweiterung von IP um

- Fehlerkorrektur (Reihenfolge und Wiederholung) durch Nummerierung der Pakete
- Ports
- “Verbindungsauf- und abbau”

TCP 1

TCP = Transmission Control Protocol

Erweiterung von IP um

- Fehlerkorrektur (Reihenfolge und Wiederholung) durch Nummerierung der Pakete
- Ports
- “Verbindungsauf- und abbau”

TCP hat eigenen Header, der innerhalb der IP-Payload liegt.

TCP 1

TCP = Transmission Control Protocol

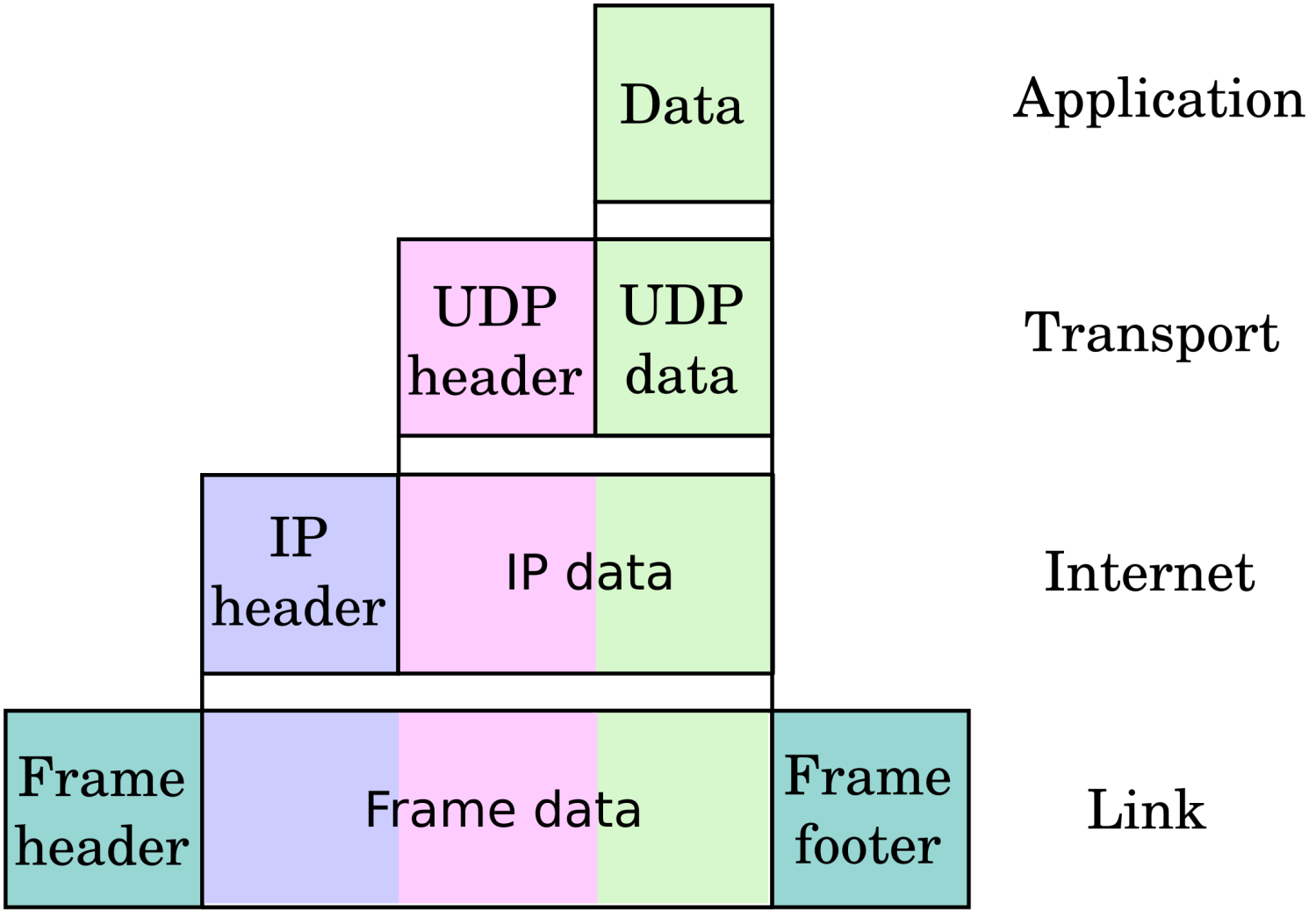
Erweiterung von IP um

- Fehlerkorrektur (Reihenfolge und Wiederholung) durch Nummerierung der Pakete
- Ports
- “Verbindungsauf- und abbau”

TCP hat eigenen Header, der innerhalb der IP-Payload liegt.

TCP regelt “alles” für die Anwendungen: Mittels TCP/IP werden Pakete verschickt bis alles vollständig und korrekt ist, erst dann erhält die Anwendung die Daten.

TCP 2



TCP 3

Nummerierung der Pakete erfolgt fortlaufend.

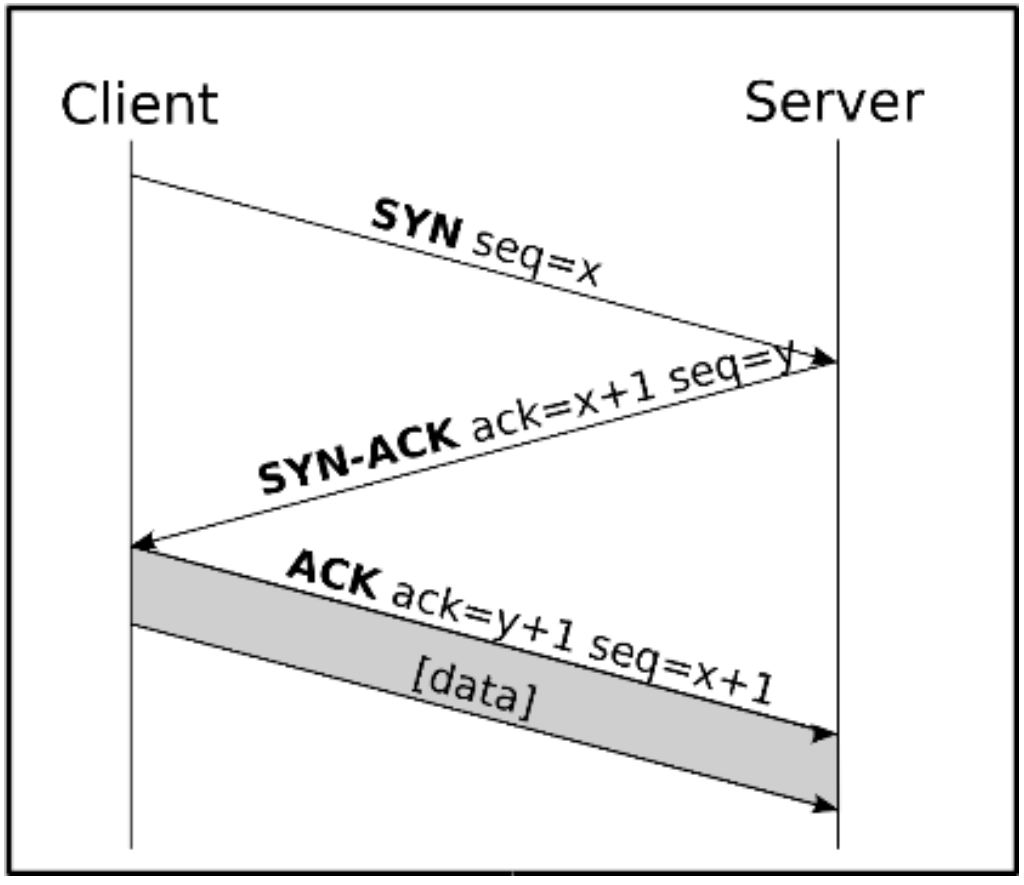
TCP 3

Nummerierung der Pakete erfolgt fortlaufend. Es werden SYN# und ACK# als Nummerierung und Bestätigung/Vorgänger geschickt.

TCP 3

Nummerierung der Pakete erfolgt fortlaufend. Es werden SYN# und ACK# als Nummerierung und Bestätigung/Vorgänger geschickt.

“Handshake”:



UDP

Ports

Jeder Rechner (adressiert durch seine IP) hat sogenannte Ports, die von Anwendungen reserviert werden.

Ports

Jeder Rechner (adressiert durch seine IP) hat sogenannte Ports, die von Anwendungen reserviert werden.

Pings

Time to live

DNS

URL Encoding

HTTP

HTTPS

SMTP, POP3, IMAP4

FTP und SFTP

ICMP

SSH und SCP

Whois

Apache und Nginx

PHP und Javascript

SQL

Curl und wget

Netcat

Telnet

SSH und SCP

Host

Whois

Ping

Nmap

Nmap Resultate

Nmap OS Erkennung

Traceroute

Wireshark

Vorgehen

Ziele

ARP

Wireshark mit ARP Spoofing

WEP-RC4

Aircrack

Session IDs

Cookies

Beispiel: PhPBB

SQL Injection

Iodine

DNS Tunneling