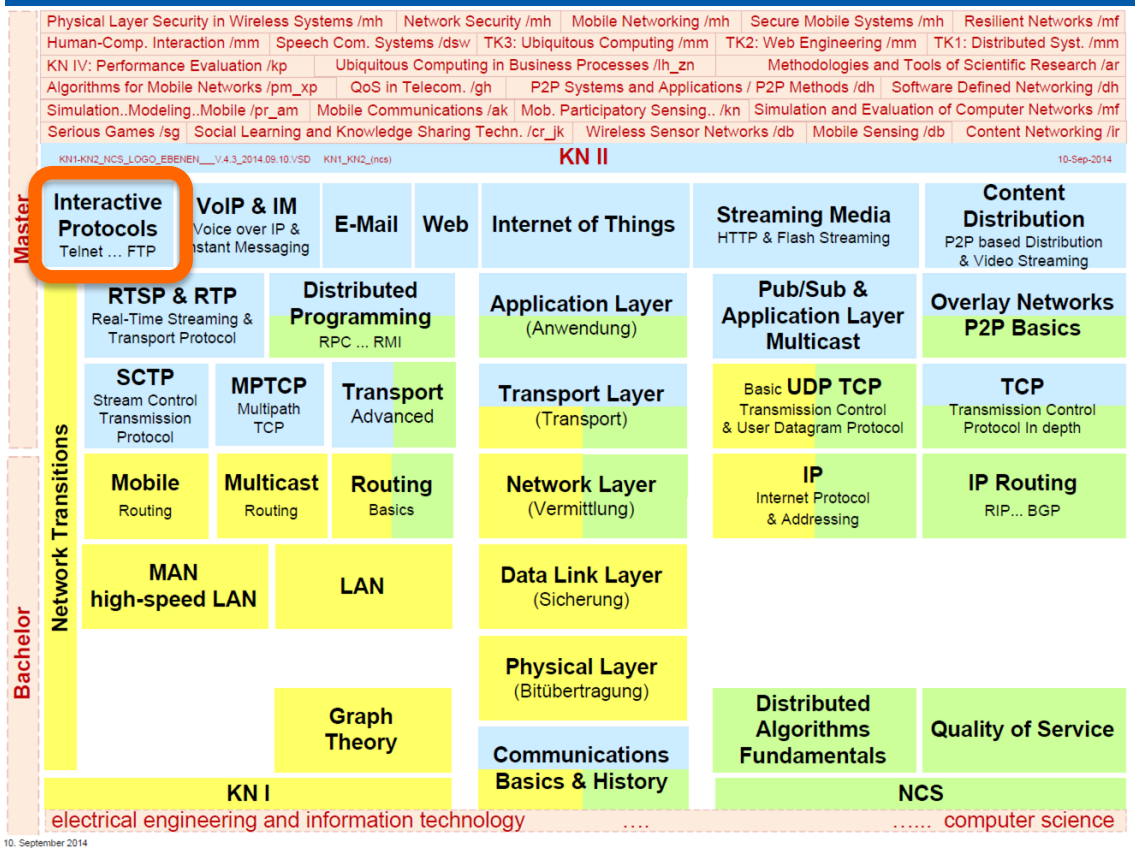
Communication Networks 2

Interactive Protocols
FTP, Telnet, SSH



Prof. Dr.-Ing. **Ralf Steinmetz**
KOM - Multimedia Communications Lab

Overview

Physical Layer Security in Wireless Systems /mh | Network Security /mh | Mobile Networking /mh | Secure Mobile Systems /mh | Resilient Networks /mf

Human-Comp. Interaction /mm | Speech Com. Systems /dsw | TK3: Ubiquitous Computing /mm | TK2: Web Engineering /mm | TK1: Distributed Syst. /mm

KN IV: Performance Evaluation /kp | Ubiquitous Computing in Business Processes /lh_zn | Methodologies and Tools of Scientific Research /ar

Algorithms for Mobile Networks /pm_xp | QoS in Telecom. /gh | P2P Systems and Applications / P2P Methods /dh | Software Defined Networking /dh

Simulation..Modeling..Mobile /pr_am | Mobile Communications /ak | Mob. Participatory Sensing.. /kn | Simulation and Evaluation of Computer Networks /mf

Serious Games /sg | Social Learning and Knowledge Sharing Techn. /cr_jk | Wireless Sensor Networks /db | Mobile Sensing /db | Content Networking /ir

KN1-KN2_NCS_LOGO_EBENEN___V.4.3_2014.09.10.VSD KN1_KN2_(ncs) **KN II** 10-Sep-2014

Master

Interactive Protocols Telnet … FTP | **VoIP & IM** Voice over IP & Instant Messaging | **E-Mail** | **Web** | **Internet of Things** | **Streaming Media** HTTP & Flash Streaming | **Content Distribution** P2P based Distribution & Video Streaming

Network Transitions

RTSP & RTP Real-Time Streaming & Transport Protocol | **Distributed Programming** RPC ... RMI | **Application Layer** (Anwendung) | **Pub/Sub & Application Layer Multicast** | **Overlay Networks P2P Basics**

SCTP Stream Control Transmission Protocol | **MPTCP** Multipath TCP | **Transport** Advanced | **Transport Layer** (Transport) | Basic **UDP TCP** Transmission Control & User Datagram Protocol | **TCP** Transmission Control Protocol In depth

Bachelor

Mobile Routing | **Multicast** Routing | **Routing** Basics | **Network Layer** (Vermittlung) | **IP** Internet Protocol & Addressing | **IP Routing** RIP... BGP

MAN high-speed LAN | **LAN** | **Data Link Layer** (Sicherung)

Physical Layer (Bitübertragung)

Graph Theory | **Distributed Algorithms Fundamentals** | **Quality of Service**

Communications Basics & History

KN I | **NCS**

electrical engineering and information technology …. …… computer science

10. September 2014

1 Basics of Remote Computing

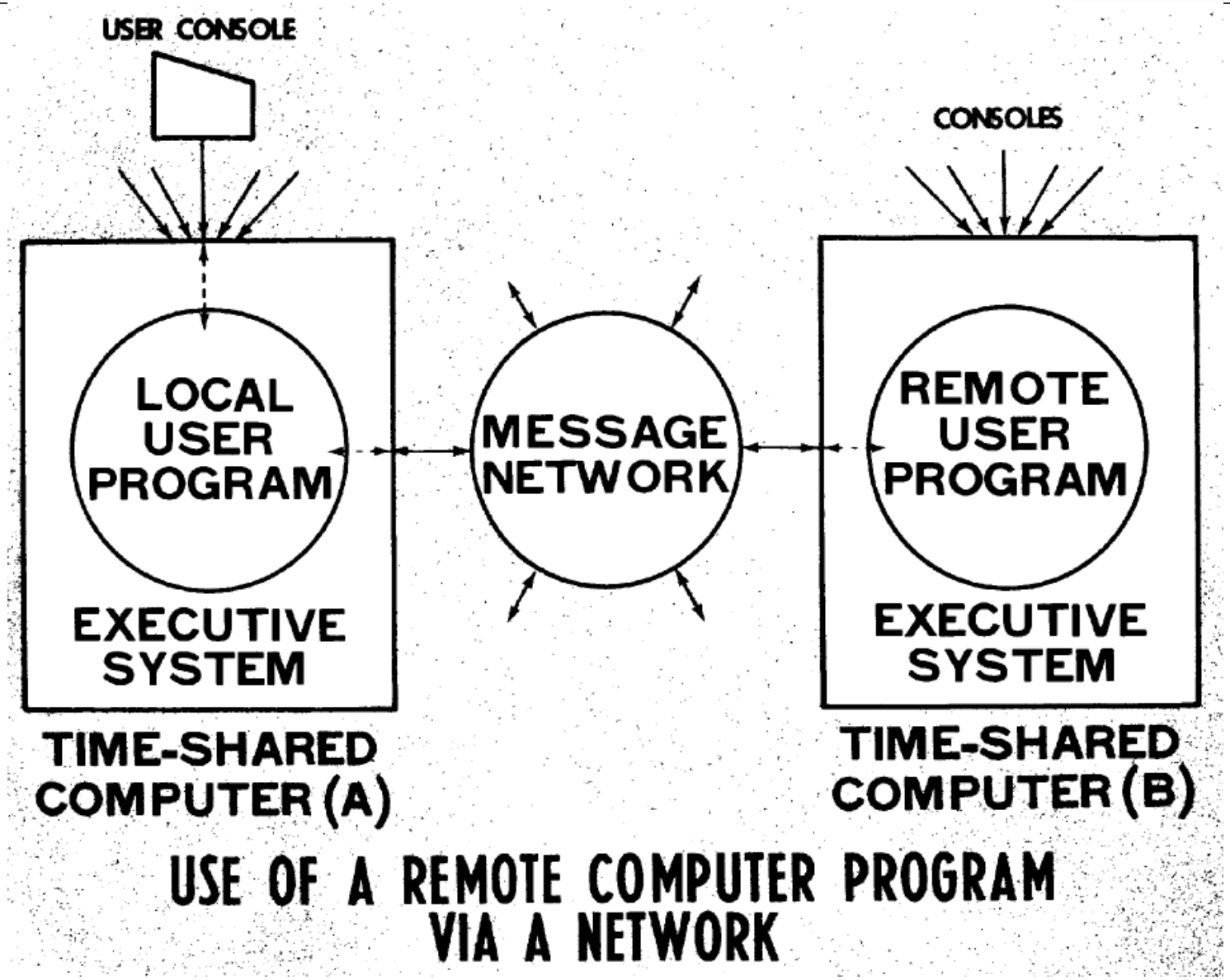


Image source: Multiple Computer Networks and Intercomputer Communication, SOSP 1967

It all began with connecting terminals to remote machines

Terminal = Endpoint

- Of computer
- Where users enter/access data

Recall: Forefather of the ARPANET

- First experiment in 1965
- In general: connect remote computer instead of local terminal

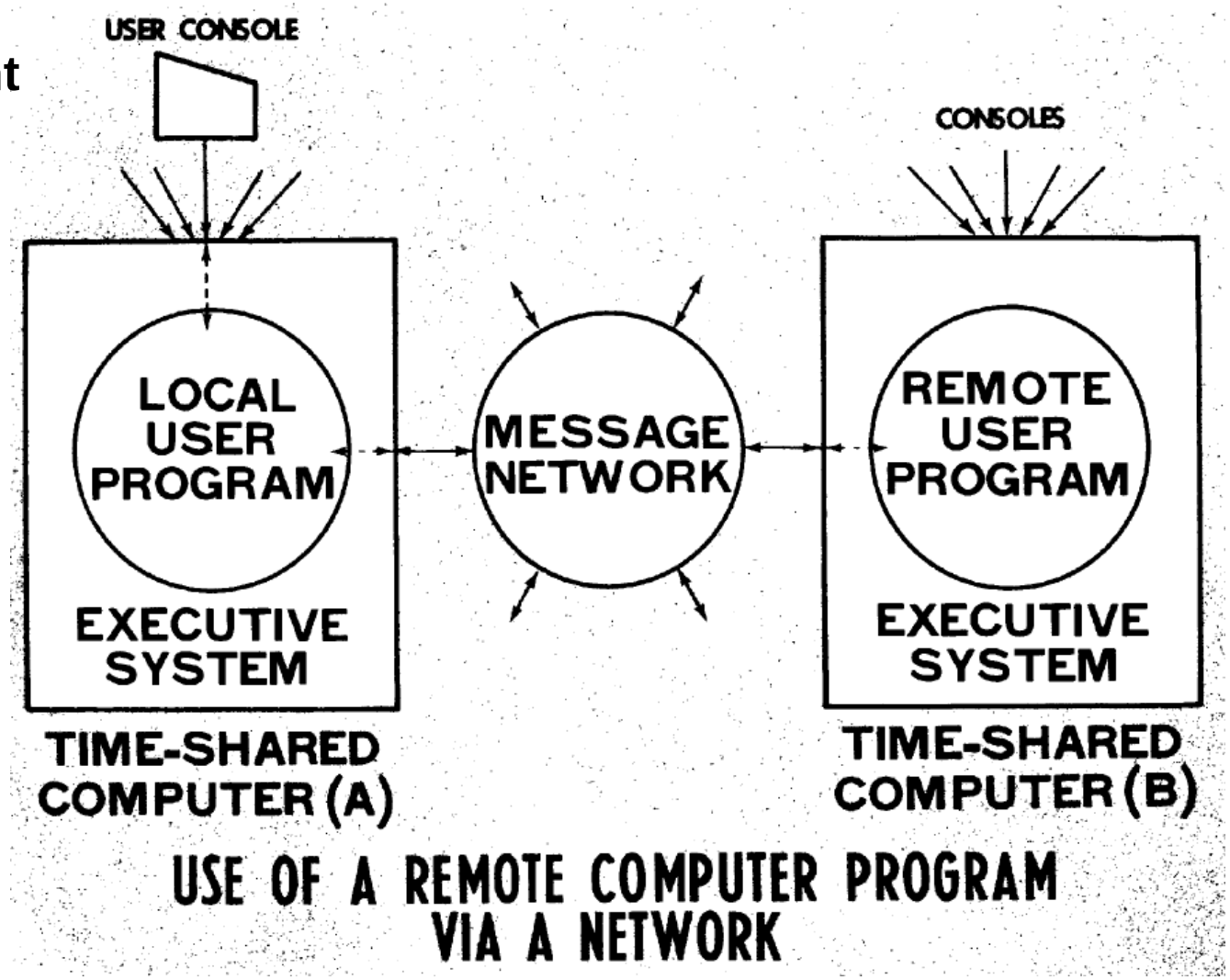


Image source: Multiple Computer Networks and Intercomputer Communication, SOSP 1967

1.2 First Remote Computing Applications

Direct usage: remote interactive terminals

- RFC 15: ‘Network Subsystem for Time Sharing Host’, September 1969
- Simple functionality provided in ‘Telnet’ subsystem (application)
  - ‘…shell program around the network system primitives, allowing a teletype or similar terminal at a remote host to function as a teletype at the serving host’
  - Open primary connection
  - Open auxiliary connection
  - Transmit over connection
  - Close connection

Indirect usage: file transfer

- RFC 114: ‘A File Transfer Protocol’, April 1971
- No direct login to remote system
- Functionality of remote system hidden by file transfer program
  - Program provides common instruction set
  - Translated to system commands at remote host

Telnet provides direct usage of remote machines

- First idea presented September 1969 in RFC 15
- Current standard version defined May 1983 in RFC 854
- More than 100 RFCs defining updates and amendments
- Today mostly replaced by Secure Shell (SSH)
- However, still in use
  - E.g. in Cisco IOS

Primary functionality: network virtual terminal

- Translates between incompatible terminal types
  - E.g. different keyboard layouts and printer (screen) capabilities
- Translates between incompatible character sets
  - Recall: computers were highly proprietary in the early times
- Offers common basic terminal functionality
  - Additional functionality can be negotiated
    - Using Terminal options

2.1 Network Virtual Terminal (NVT)

Network virtual terminal is bi-directional character-oriented device

- Consisting of printer (today: screen) and keyboard
- Printer handles incoming data
- Keyboard produces outgoing data

Character set: 7 bit US ASCII

- 95 printable characters:
  0123456789!"#$%&'()*+,-./:;<=>@[\]^_`{|}~
  ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
- 33 non-printable control characters
  - For controlling printer, not (!) terminal
  - E.g. backspace, delete, carriage return, line feed
- Conversion from/to NVT character set performed by Telnet application

However, 8 bit byte representation used

- Required e.g. for transferring binary data

Example: starting Telnet session to remote host

- From MAC OS machine at 192.168.178.20
- To Linux machine at 130.83.125.13

Andre — akoenig@glab013: ~ — telnet — 80×24

```
Andres-MacBook-Pro:~ Andre$ telnet -l akoenig 130.83.125.13
Trying 130.83.125.13...
Connected to glab013.g-lab.tu-darmstadt.de.
Escape character is '^]'.
Password:
Last login: Sat Dec 10 18:46:08 CET 2011 from p4FC73FC1.dip.t-dialin.net on pts/
1
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-13-generic i686)

 * Documentation:  https://help.ubuntu.com/

akoenig@glab013:~$
```

Network Virtual Terminal (NVT)

NVT understands minimal set of Terminal commands

- Implemented on all systems to provide common functionality
- Represented above 7 bit ASCII character range

Character stuffing used to indicate character is control character

- Required because binary data may contain command characters
  - E.g. when transferring file
- Command code 255 (0xff, 11111111) is stuffing character
  - Interpret As Command (IAC) command

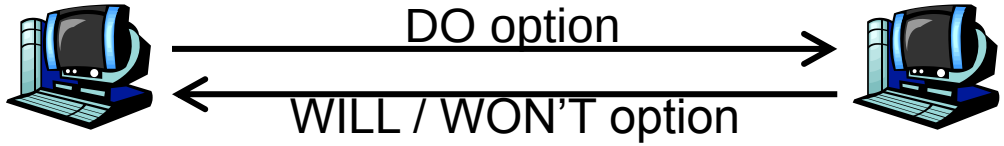
E.g. commands to negotiate additional terminal options

- WILL command (0xfb): desire / request for using option
- WON’T command (0xfc): refusal to use option
- DO command (0xfd): request to / confirmation for using option
- DON’t command (0xfe): request to / confirmation for stop using option

NVT option negotiation process can be request or indication

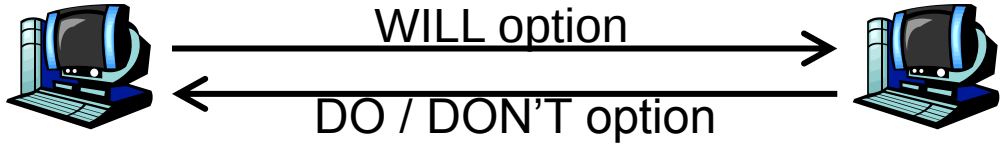
Option request

- I.e. requesting host wants remote host to turn on an option
- Request by DO command
- Reply is WILL or WON’T command



Option indication

- I.e. indicating host wants to locally use an option
- Indication by WILL command
- Reply is DO or DON’T command



Network Virtual Terminal (NVT)

TECHNISCHE UNIVERSITÄT DARMSTADT

Currently 53 NVT options specified

- See http://www.iana.org/assignments/telnet-options

E.g. echo option (0x01)

- Defined May 1983 in RFC 857
- Turn on / off echoes of characters received sent back over the network

E.g. linemode option (0x22)

- Defined October 1990 in RFC 1184
- Turn on / off transmitting whole lines instead of single characters

E.g. negotiate about window size option (0x1f)

- Defined October 1988 in RFC 1073
- Turn on / off variable window size for NVT
- Includes suboption for announcing window size

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. linemode option

- Option not enabled after connection setup
- Client indicates linemode

```
▷ Frame 6: 96 bytes on wire (768 bits), 96 bytes captured (768 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931374, Ack: 233653059, Len: 30
▽ Telnet
    Command: Will Linemode

0050  20 ff fb 21 ff fb 22 ff  fb 27 ff fd 05 ff fb 23    ..!..". .'.....#
```

- Server refuses linemode

```
▷ Frame 10: 108 bytes on wire (864 bits), 108 bytes captured (864 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49993 (49993), Seq: 233653071, Ack: 2849931404, Len: 42
▽ Telnet
    Command: Don't Linemode

0040  d4 e9 ff fe 25 ff fb 03  ff fd 1f ff fd 21 ff fe    ....%... .....!..
0050  22 ff fb 05 ff fa 20 01  ff f0 ff fa 23 01 ff f0    ".....  . ....#...
```

Network Virtual Terminal (NVT)

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. echo option

- Option negotiation after connection setup
- Server requests client to produce its own echo
  - I.e. client should display typed characters

```
▷ Frame 13: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49993 (49993), Seq: 233653113, Ack: 2849931534, Len: 3
▽ Telnet
    Command: Do Echo
0040  d5 00 ff fd 01                                        ......
```

- Client refuses to produce own echo

```
▷ Frame 15: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931534, Ack: 233653116, Len: 3
▽ Telnet
    Command: Won't Echo
0040  98 a0 ff fc 01                                        ......
```

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. echo option (cont'd)

- Server announces it will send echoes to client

```
▷ Frame 16: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49993 (49993), Seq: 233653116, Ack: 2849931537, Len: 3
▽ Telnet
    Command: Will Echo

0040   d5 1b ff fb 01                                        ......
```

- Client confirms that server should send echoes

```
▷ Frame 18: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931537, Ack: 233653119, Len: 3
▽ Telnet
    Command: Do Echo

0040   98 a7 ff fd 01                                        .....
```

TECHNISCHE UNIVERSITÄT DARMSTADT

Result of option negotiation

- No linemode, no local echoes
- Each character transmitted directly
- Each character echoed from server to client
- E.g. issuing ‘top’ command (show server load)
  - 4 messages from client to server (t, o, p, return)

```
▷ Frame 228: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931559, Ack: 233653411, Len: 1
▽ Telnet
    Data: t
```

  - 4 messages from server to client (t, o, p, return)

```
▷ Frame 229: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49993 (49993), Seq: 233653411, Ack: 2849931560, Len: 1
▽ Telnet
    Data: t
```

Network Virtual Terminal (NVT)

E.g. negotiate about window size option

- Manual change of Telnet window size
- From 80 columns
- To 81 columns

Andre — akoenig@glab013: ~ — telnet — 80×24

```
Andres-MacBook-Pro:~ Andre$ telnet -l akoenig 130.83.125.13
Trying 130.83.125.13...
Connected to glab013.g-lab.tu-darmstadt.de.
Escape character is '^]'.
Password:
Last login: Sat Dec 10 18:46:08 CET 2011 from p4FC73FC1.dip.t-dialin.net on pts/
1
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-13-generic i686)

 * Documentation:  https://help.ubuntu.com/

akoenig@glab013:~$
```

Andre — akoenig@glab013: ~ — telnet — 81×24

```
Andres-MacBook-Pro:~ Andre$ telnet -l akoenig 130.83.125.13
Trying 130.83.125.13...
Connected to glab013.g-lab.tu-darmstadt.de.
Escape character is '^]'.
Password:
Last login: Sat Dec 10 18:46:08 CET 2011 from p4FC73FC1.dip.t-dialin.net on pts/1
Welcome to Ubuntu 11.10 (GNU/Linux 3.0.0-13-generic i686)

 * Documentation:  https://help.ubuntu.com/

akoenig@glab013:~$
```

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. negotiate about window size option

- Option enabled after connection setup
- Indication by client

```
▷ Frame 6: 96 bytes on wire (768 bits), 96 bytes captured (768 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931374, Ack: 233653059, Len: 30
▽ Telnet
    Command: Will Negotiate About Window Size
0040  98 89 ff fb 25 ff fd 03  ff fb 18 ff fb 1f ff fb   ....%... ........
```

- Confirmation by server

```
▷ Frame 10: 108 bytes on wire (864 bits), 108 bytes captured (864 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 23 (23), Dst Port: 49993 (49993), Seq: 233653071, Ack: 2849931404, Len: 42
▽ Telnet
    Command: Do Negotiate About Window Size
0040  d4 e9 ff fe 25 ff fb 03  ff fd 1f ff fd 21 ff fe   ....%... .....!..
```

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. negotiate about window size option

- Change of window size announced in suboption
- Sent from client to server

```
▷ Frame 182: 75 bytes on wire (600 bits), 75 bytes captured (600 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931550, Ack: 233653364, Len: 9
▽ Telnet
  ▽ Suboption Begin: Negotiate About Window Size
      Width: 81
      Height: 24
    Command: Suboption End

0040  9e a5 ff fa 1f 00 51 00  18 ff f0                    ......Q. ...
```

2.2 NVT Control Functions

Network virtual terminal control functions

- User accessible NVT commands
- Options for controlling terminal
- Hide specific control functions of remote terminal

Also represented above 7 bit ASCII character range

Also preceded by IAC command

Can be sent by switching to Telnet command mode

- By entering escape sequence announced during connection setup
  - Usually CTRL +]

E.g. interrupt process command

- Control code 244 (0xf4, 11110100)
- Interrupt process on unix: CTRL + C
- But CTRL + C on e.g. windows: copy to clipboard
- Solution: send interrupt process command in command mode

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. Interrupt process command

- Remote program started (here: top)
- Process got hung

Andre — akoenig@glab013: ~ — telnet — 81×24

```
top - 18:49:33 up 4 days,  2:32,  2 users,  load average: 0.02, 0.04, 0.05
Tasks: 126 total,   1 running, 125 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.3%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    508388k total,   490900k used,    17488k free,   149220k buffers
Swap:   407548k total,     6540k used,   401008k free,   139632k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
17189 akoenig   20   0  2820 1140  856 R  0.3  0.2   0:00.03 top
    1 root      20   0  3332 1708 1212 S  0.0  0.3   0:02.61 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.03 kthreadd
    3 root      20   0     0    0    0 S  0.0  0.0   0:02.65 ksoftirqd/0
    5 root      20   0     0    0    0 S  0.0  0.0   0:00.31 kworker/u:0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    7 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 cpuset
    8 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 khelper
    9 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 netns
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.86 sync_supers
   11 root      20   0     0    0    0 S  0.0  0.0   0:00.01 bdi-default
   12 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kintegrityd
   13 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kblockd
   14 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 ata_sff
   15 root      20   0     0    0    0 S  0.0  0.0   0:00.06 khubd
   16 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 md
   18 root      20   0     0    0    0 S  0.0  0.0   0:00.27 kworker/u:1
```

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. Interrupt process command

- Enter Telnet command mode
- Send interrupt process command

Andre — akoenig@glab013: ~ — telnet — 81×24

```
top - 18:49:57 up 4 days,  2:32,  2 users,  load average: 0.01, 0.03, 0.05
Tasks: 126 total,   1 running, 125 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.3%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    508388k total,   490900k used,    17488k free,   149220k buffers
Swap:   407548k total,     6540k used,   401008k free,   139632k cached

telnet> send ip
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1736 akoenig   20   0 56168  10m 8432 S  0.3  2.1   0:07.25 update-notifier
17189 akoenig   20   0  2820 1140  856 R  0.3  0.2   0:00.07 top
    1 root      20   0  3332 1708 1212 S  0.0  0.3   0:02.61 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.03 kthreadd
    3 root      20   0     0    0    0 S  0.0  0.0   0:02.65 ksoftirqd/0
    5 root      20   0     0    0    0 S  0.0  0.0   0:00.31 kworker/u:0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    7 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 cpuset
    8 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 khelper
    9 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 netns
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.86 sync_supers
   11 root      20   0     0    0    0 S  0.0  0.0   0:00.01 bdi-default
   12 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kintegrityd
   13 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kblockd
   14 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 ata_sff
   15 root      20   0     0    0    0 S  0.0  0.0   0:00.06 khubd
   16 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 md
```

NVT Control Functions

E.g. Interrupt process command

- Enter Telnet command mode
- Send interrupt process command

```
835 208.656921   192.168.178.20   130.83.125.13    TELNET        68 Telnet Data ...
```

```
▷ Frame 835: 68 bytes on wire (544 bits), 68 bytes captured (544 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931564, Ack: 233750127, Len: 2
▽ Telnet
    Command: Interrupt Process

0040   60 65 ff f4                                          `e..
```

TECHNISCHE UNIVERSITÄT DARMSTADT

E.g. Interrupt process command

- Result: process ended

Andre — akoenig@glab013: ~ — telnet — 81×24

```
Tasks: 126 total,   1 running, 125 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.0%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    508388k total,   490900k used,    17488k free,   149256k buffers
Swap:   407548k total,     6540k used,   401008k free,   139636k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
17598 akoenig   20   0  2820 1140  856 R  0.3  0.2   0:00.02 top
    1 root      20   0  3332 1708 1212 S  0.0  0.3   0:02.61 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.03 kthreadd
    3 root      20   0     0    0    0 S  0.0  0.0   0:02.65 ksoftirqd/0
    5 root      20   0     0    0    0 S  0.0  0.0   0:00.31 kworker/u:0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    7 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 cpuset
    8 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 khelper
    9 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 netns
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.86 sync_supers
   11 root      20   0     0    0    0 S  0.0  0.0   0:00.01 bdi-default
   12 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kintegrityd
   13 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kblockd
   14 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 ata_sff
   15 root      20   0     0    0    0 S  0.0  0.0   0:00.06 khubd
   16 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 md
   18 root      20   0     0    0    0 S  0.0  0.0   0:00.27 kworker/u:1
akoenig@glab013:~$
```

TECHNISCHE UNIVERSITÄT DARMSTADT

Telnet by default is plaintext protocol

- Encryption only optional
  - Implementation not mandatory
- Passive attacks easily possible
  - E.g. capturing password (‘password’) during login

```
▷ Frame 22: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931540, Ack: 233653129, Len: 1
▽ Telnet
    Data: p
0040  98 b7 70                                                ..p
```

```
▷ Frame 24: 67 bytes on wire (536 bits), 67 bytes captured (536 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 49993 (49993), Dst Port: 23 (23), Seq: 2849931541, Ack: 233653129, Len: 1
▽ Telnet
    Data: a
0040  9b 2a 61                                                .*a
```

TECHNISCHE UNIVERSITÄT DARMSTADT

Secure shell

- Version 1 developed 1995 at Helsinki University
  - As replacement for unsecure Telnet
  - Motivated by a password sniffing attack
- Version 2 specified January 2006 in RFCs 4251-4254

Secure shell provides

- A network virtual terminal
  - Extended functionality compared to Telnet
- Host and user authentication
- Encrypted communication

Secure shell consists of

- SSH Transport Layer Protocol
- SSH Authentication Protocol
- SSH Connection Protocol

SSH Connection Protocol
Interactive shell sessions, port forwarding

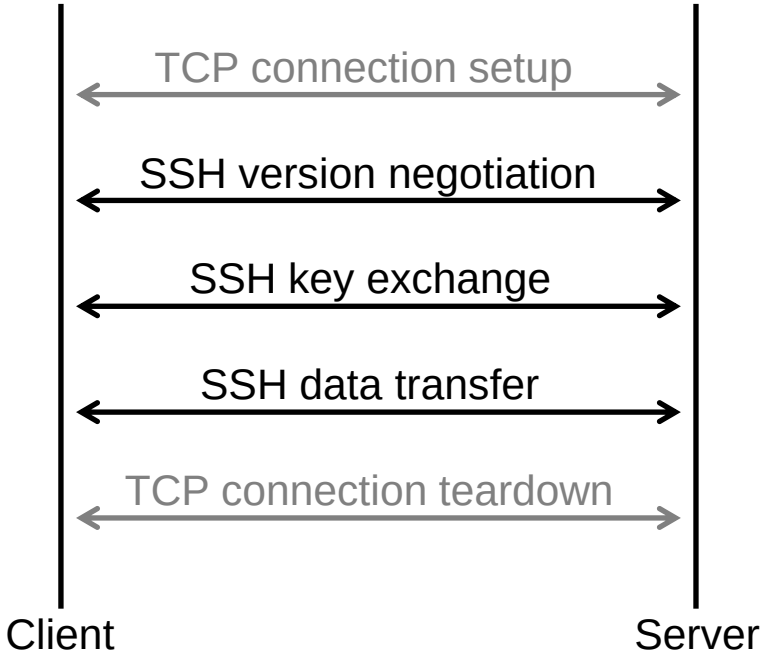
SSH User Authentication Protocol
Client-side user authentication

SSH Transport Layer Protocol
Server authentication, confidentiality, integrity

Common reliable transport protocol (e.g. TCP)

Task: set up secure tunnel between client and server

- Step 1: trigger TCP connection setup
  - Server listens on TCP port 22
- Step 2: negotiate SSH version
- Step 3: key exchange
  - E.g. based on Diffie-Hellmann
- Step 4: data transfer
  - Including SSH user authentication
- Step 5: close TCP connection



Task: authenticate server to client

- Different models described

- Global certification authority
  - Server provides certificate obtained from CA
  - Authenticity verified automatically

- Local database at client (e.g. ~/.ssh/known_hosts)
  - Server provides its public key during key exchange
  - Database associates host name with public host key
  - Authenticity verified by client's administrator
    - E.g. signature of public host key published on web site or checked on phone

SSH User Authentication Protocol

- Requires a connection offering confidentiality and integrity
  - Offered by SSH Transport Layer Protocol

Three different authentication methods supported

- Password-based authentication
  - Should be supported by all SSH implementations
  - User authenticates by providing password
  - User/password tuples maintained by server

- Public key-based authentication
  - Must be supported by all SSH implementations
  - User authenticates by providing signature produced with private key
  - Server verifies signature
  - User/public key tuples can be (as for host auth. in SSH Transport Layer Protocol)
    - Maintained locally on server
    - Obtained by certification authority

- Host-based authentication
  - Like public key authentication
  - On host-level, not on user-level (as for host auth. in SSH Transport Layer Protocol)

Services provided by SSH Connection Protocol

- Interactive sessions
  - Network virtual terminal (compare: telnet)

- Non-interactive execution of remote commands
  - Command specified in ssh command line

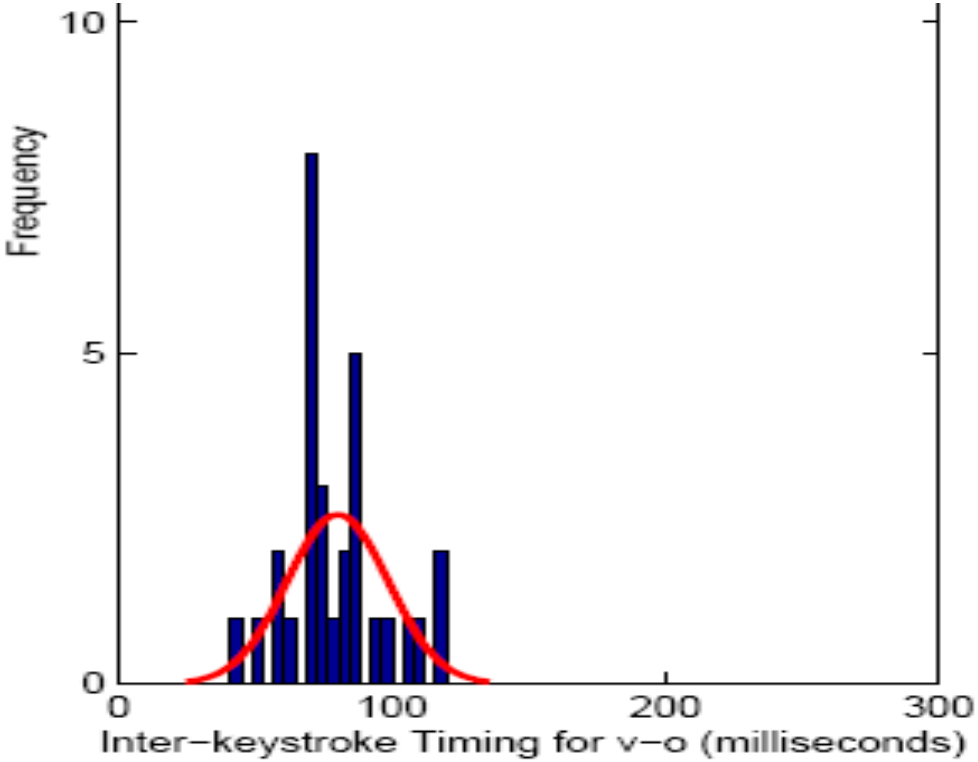
```
Andres-MacBook-Pro:~ Andre$ ssh akoenig@130.83.125.13 who
akoenig  pts/0        Dec  6 16:19 (:0)
Andres-MacBook-Pro:~ Andre$
```

- Forwarding of TCP ports through SSH tunnel
  - Local: from client to server
  - Remote: from server to client
  - Used e.g. to traverse firewalls for remote desktop connections

- Forwarding X11 connections
  - Display programs running on server locally

Attacks on SSH possible despite encryption and authentication

- E.g. statistical side channel attacks
- SSH transmits typed character directly
  - Like Telnet with linemode disabled
- Inter-keystroke time can be measured
  - Patterns can be analyzed



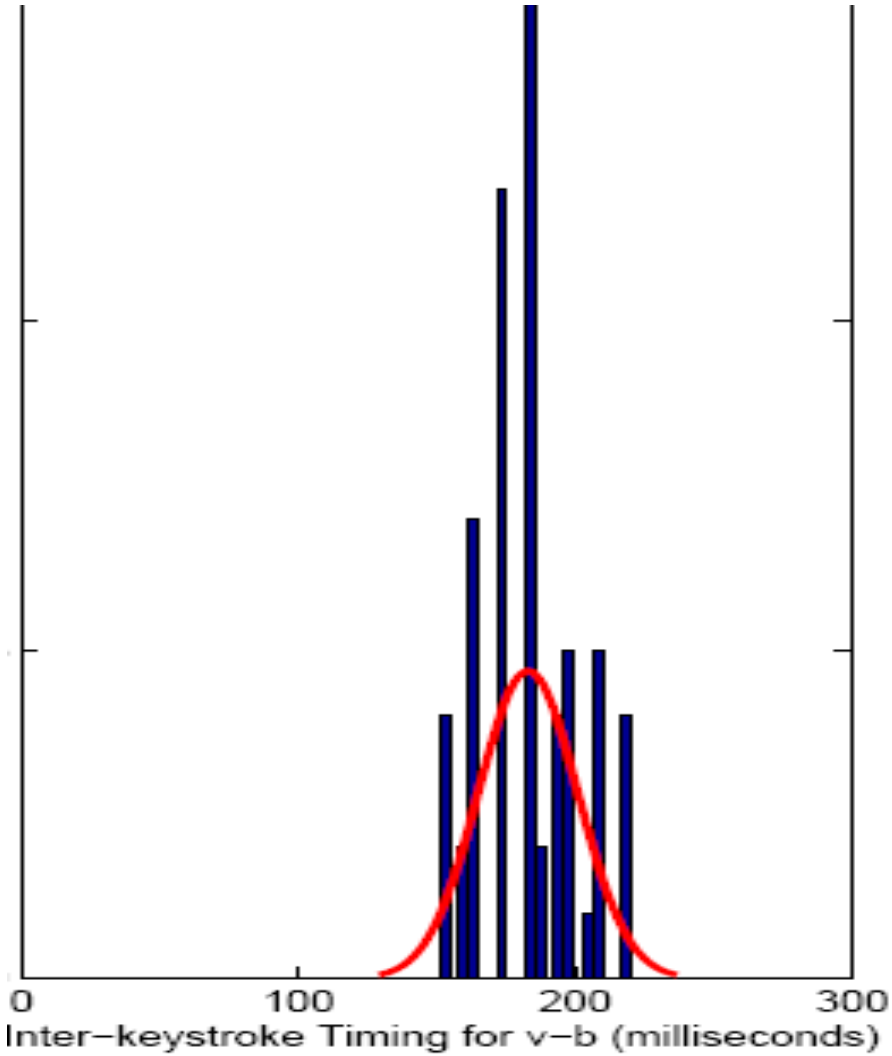
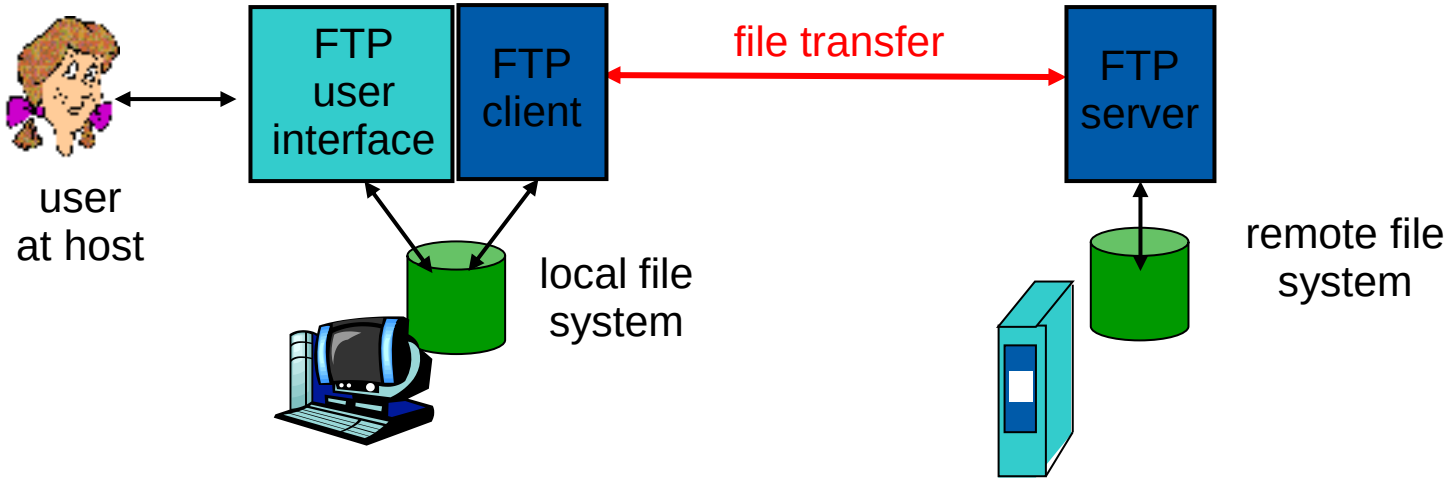


Image source: Tam Ngo, Steve Licking: Secure Shell - SSH

TECHNISCHE UNIVERSITÄT DARMSTADT

File Transfer Protocol (FTP)

- Belongs to class of indirect remote computing applications
  - Instruction set for remote file system hidden
    - By commands of FTP client
    - Note: different from commands of protocol itself
- First presented April 1971 in RFC 114
- Current version as of October 1985, RFC 959
- Server listens on port 21 for client connections



Sample FTP session

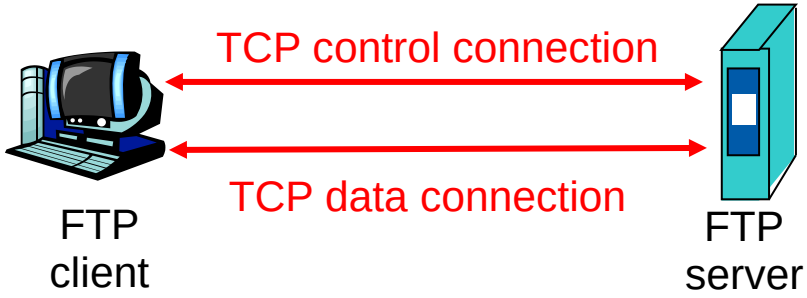
- Established by client 192.168.178.20
  - With server 130.83.125.13
- Frontend: standard Unix text-based FPT client
- Other client e.g. Filezilla with graphical UI

Andre — akoenig@glab013: ~ — bash — 80×24

```
Andres-MacBook-Pro:~ Andre$ ftp
ftp> open 130.83.125.13
Connected to 130.83.125.13.
220 ProFTPD 1.3.4rc2 Server (Debian) [::ffff:130.83.125.13]
Name (130.83.125.13:Andre): akoenig
331 Password required for akoenig
Password:
230 User akoenig logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> send Desktop/telnet.ppt
local: Desktop/telnet.ppt remote: Desktop/telnet.ppt
229 Entering Extended Passive Mode (|||35364|)
150 Opening BINARY mode data connection for Desktop/telnet.ppt
100% |***********************************| 50176      284.83 MiB/s    00:00 ETA
226 Transfer complete
50176 bytes sent in 00:00 (107.69 KiB/s)
ftp> exit
221 Goodbye.
Andres-MacBook-Pro:~ Andre$
```

FTP uses two TCP connections between client and server

- Control connection
  - Used to transmit ASCII commands from client to server
    - E.g. for user log in and file browsing
    - USER xy - log in as user ‘username’
    - PASS password - send user password
    - LIST - return list of file in current directory
    - RETR - filename retrieves (gets) file
    - STOR - filename stores (puts) file onto remote host
  - Used to transmit reply codes from server to client
    - 220 - ready for new user
    - 331 - username OK, password required
    - 230 - user logged in
    - 425 - unable to build data connection

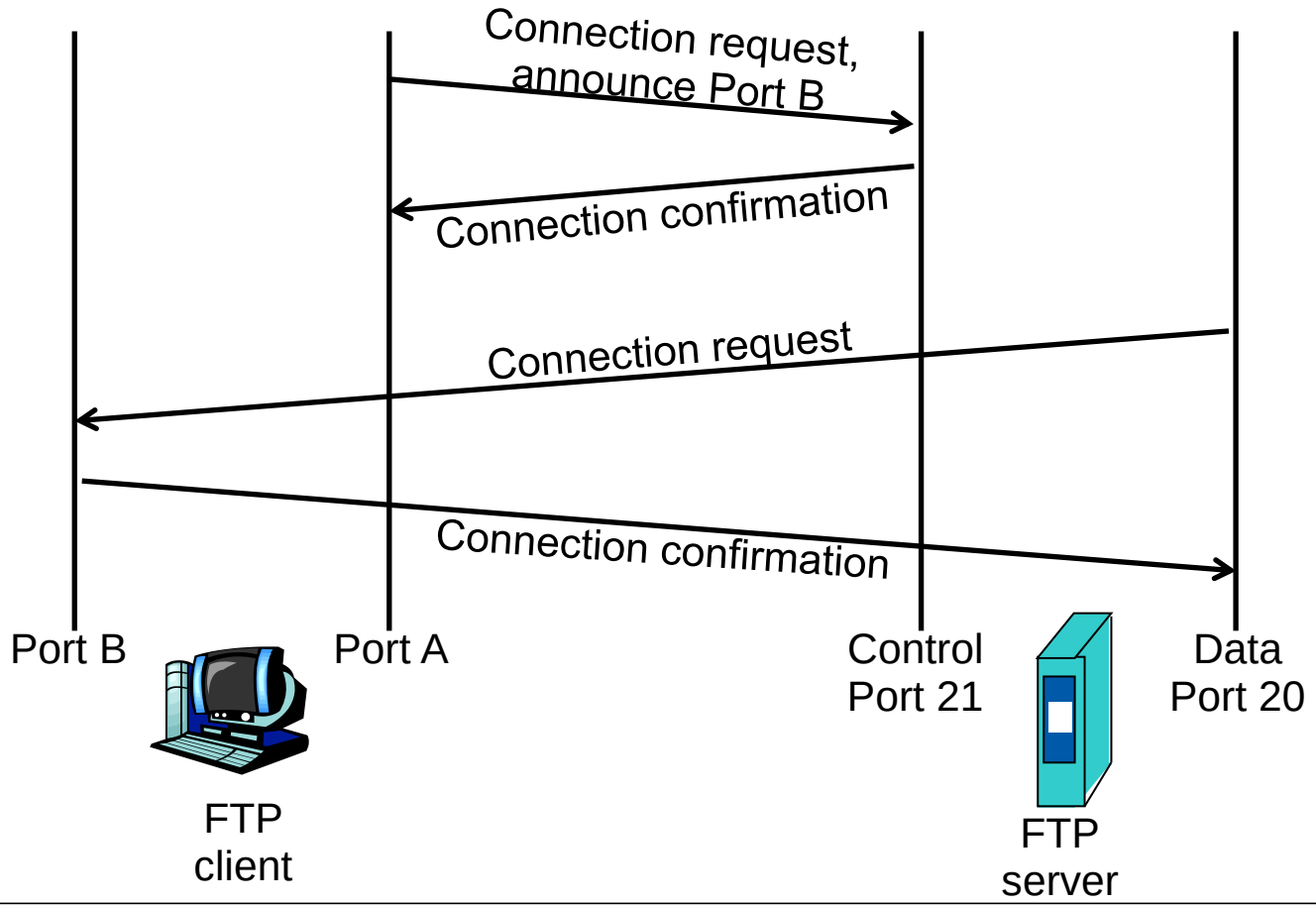


- Data connection
  - Used to send or receive files (binary data)
  - Established on a per file basis

FTP Connection Model

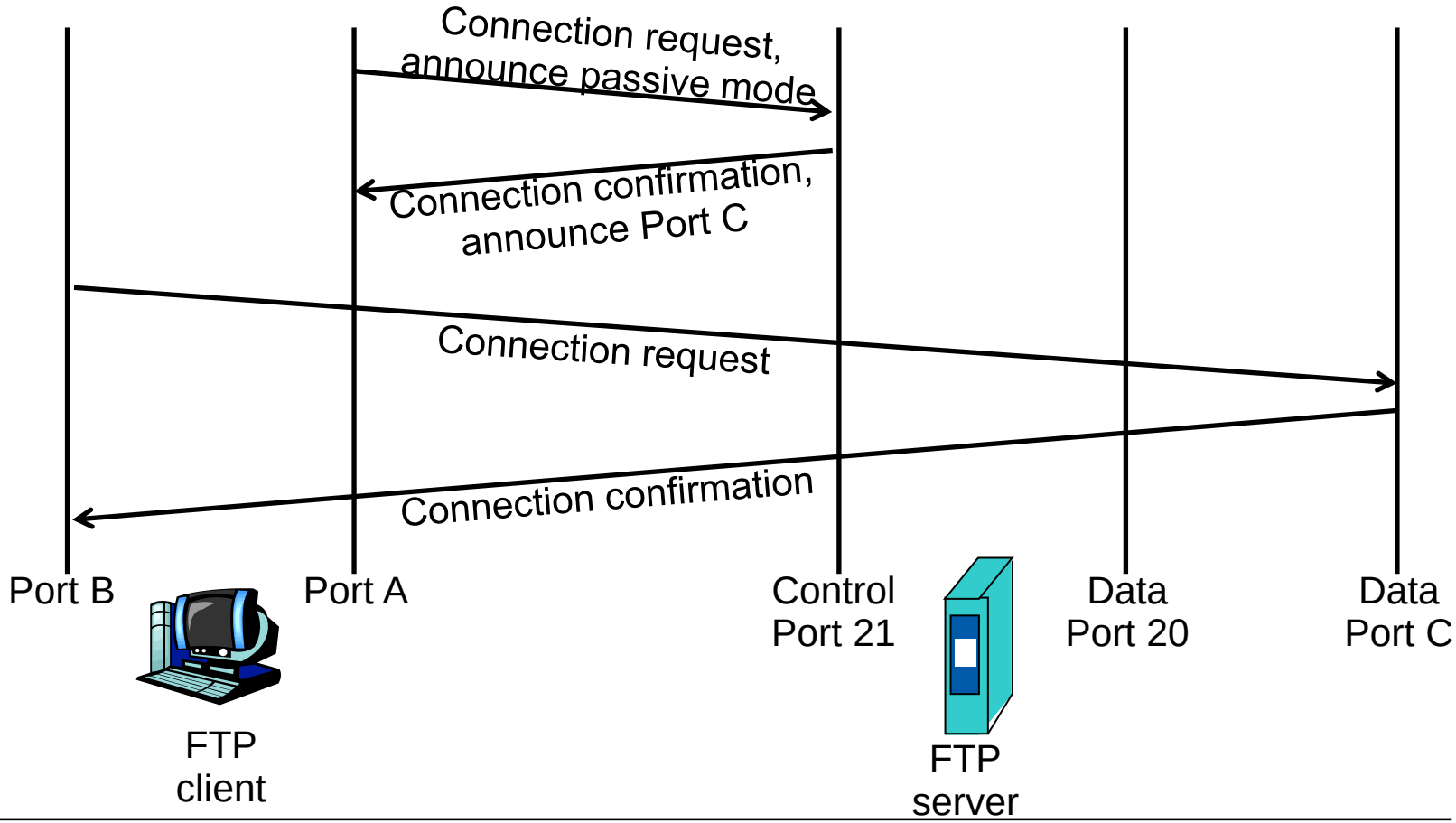
FTP active mode

- Data connection established by server
- Client announces port for data connection
  - During connection setup



FTP passive mode

- Data connection established by client
  - Required e.g. if client is behind NAT router
- Server announces port for data connection



TECHNISCHE UNIVERSITÄT DARMSTADT

Sample FTP session

- Client 192.168.178.20
- Server 130.83.125.13
- Commands performe
  - Log in
  - Send file
  - Log out

Andre — akoenig@glab013: ~ — bash — 80×24

```
Andres-MacBook-Pro:~ Andre$ ftp
ftp> open 130.83.125.13
Connected to 130.83.125.13.
220 ProFTPD 1.3.4rc2 Server (Debian) [::ffff:130.83.125.13]
Name (130.83.125.13:Andre): akoenig
331 Password required for akoenig
Password:
230 User akoenig logged in
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> send Desktop/telnet.ppt
local: Desktop/telnet.ppt remote: Desktop/telnet.ppt
229 Entering Extended Passive Mode (|||35364|)
150 Opening BINARY mode data connection for Desktop/telnet.ppt
100% |***********************************| 50176      284.83 MiB/s    00:00 ETA
226 Transfer complete
50176 bytes sent in 00:00 (107.69 KiB/s)
ftp> exit
221 Goodbye.
Andres-MacBook-Pro:~ Andre$
```

FTP Wireshark Example

Connection establishment

- Client establishes TCP connection
- Server reacts with log in prompt
  - Code 220 - ready for new user

3	0.004241	192.168.178.20	130.83.125.13	TCP	78	51665 > 21 [SYN]
4	0.029519	130.83.125.13	192.168.178.20	TCP	74	21 > 51665 [SYN, ACK]
5	0.029702	192.168.178.20	130.83.125.13	TCP	66	51665 > 21 [ACK]
6	0.063373	130.83.125.13	192.168.178.20	FTP	127	Response: 220 ProFTPD 1.3.4rc2 Server (Debian)

```
▷ Frame 6: 127 bytes on wire (1016 bits), 127 bytes captured (1016 bits)
▷ Ethernet II, Src: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8), Dst: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c)
▷ Internet Protocol Version 4, Src: 130.83.125.13 (130.83.125.13), Dst: 192.168.178.20 (192.168.178.20)
▷ Transmission Control Protocol, Src Port: 21 (21), Dst Port: 51665 (51665), Seq: 1617093517, Ack: 173485341, Len: 61
▽ File Transfer Protocol (FTP)
  ▽ 220 ProFTPD 1.3.4rc2 Server (Debian) [::ffff:130.83.125.13]\r\n
      Response code: Service ready for new user (220)
      Response arg: ProFTPD 1.3.4rc2 Server (Debian) [::ffff:130.83.125.13]

0040  af 1f 32 32 30 20 50 72  6f 46 54 50 44 20 31 2e   ..220 Pr oFTPD 1.
0050  33 2e 34 72 63 32 20 53  65 72 76 65 72 20 28 44   3.4rc2 S erver (D
0060  65 62 69 61 6e 29 20 5b  3a 3a 66 66 66 66 3a 31   ebian) [ ::ffff:1
0070  33 30 2e 38 33 2e 31 32  35 2e 31 33 5d 0d 0a      30.83.12 5.13]..
```

FTP Wireshark Example

User log in

- Communication via control connection
- Plaintext ASCII protocol → security!

8	3.894720	192.168.178.20	130.83.125.13	FTP	80	Request: USER akoenig
10	3.925291	130.83.125.13	192.168.178.20	FTP	101	Response: 331 Password required for akoenig
16	7.670608	192.168.178.20	130.83.125.13	FTP	81	Request: PASS password
18	7.792428	130.83.125.13	192.168.178.20	FTP	94	Response: 230 User akoenig logged in

```
▷ Frame 16: 81 bytes on wire (648 bits), 81 bytes captured (648 bits)
▷ Ethernet II, Src: 8c:7b:9d:d6:dd:3c (8c:7b:9d:d6:dd:3c), Dst: bc:05:43:be:0f:a8 (bc:05:43:be:0f:a8)
▷ Internet Protocol Version 4, Src: 192.168.178.20 (192.168.178.20), Dst: 130.83.125.13 (130.83.125.13)
▷ Transmission Control Protocol, Src Port: 51665 (51665), Dst Port: 21 (21), Seq: 173485355, Ack: 1617093613, Len: 15
▽ File Transfer Protocol (FTP)
  ▽ PASS password\r\n
      Request command: PASS
      Request arg: password

0040  5d d5 50 41 53 53 20 70  61 73 73 77 6f 72 64 0d   ].PASS p assword.
```

FTP Wireshark Example

Client sends file to server

- Client announces passive mode
- Server sends port for establishing data connection
- Client establishes connection and sends file

42	16.452002	192.168.178.20	130.83.125.13	FTP	72	Request: EPSV
43	16.480786	130.83.125.13	192.168.178.20	FTP	114	Response: 229 Entering Extended Passive Mode (|||35364|)
44	16.480958	192.168.178.20	130.83.125.13	TCP	66	51665 > 21 [ACK] Seq=173485401 Ack=1617094050
45	16.481290	192.168.178.20	130.83.125.13	TCP	78	51666 > 35364 [SYN] Seq=2441503292
46	16.508052	130.83.125.13	192.168.178.20	TCP	74	35364 > 51666 [SYN, ACK] Seq=3683781746 Ack=2441503293
47	16.508172	192.168.178.20	130.83.125.13	TCP	66	51666 > 35364 [ACK] Seq=2441503293 Ack=3683781747
48	16.508282	192.168.178.20	130.83.125.13	FTP	91	Request: STOR Desktop/telnet.ppt
49	16.535217	130.83.125.13	192.168.178.20	FTP	130	Response: 150 Opening BINARY mode data connection
50	16.535318	192.168.178.20	130.83.125.13	TCP	66	51665 > 21 [ACK] Seq=173485426 Ack=1617094114
51	16.535609	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
52	16.535613	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
53	16.535614	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
54	16.573462	130.83.125.13	192.168.178.20	TCP	66	35364 > 51666 [ACK] Seq=3683781747 Ack=2441504661

TECHNISCHE UNIVERSITÄT DARMSTADT

Client sends file to server (cont'd)

- Client closes data connection after sending file
- Control connection remains open
- Server sends transfer complete message (code 226)
  - Via control connection

100	16.744431	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
101	16.744488	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
102	16.754565	130.83.125.13	192.168.178.20	TCP	66	35364 > 51666 [ACK]
103	16.754630	192.168.178.20	130.83.125.13	FTP-DATA	1434	FTP Data: 1368 bytes
104	16.754680	192.168.178.20	130.83.125.13	FTP-DATA	994	FTP Data: 928 bytes

TCP fin flag set in this message

136	16.989023	192.168.178.20	130.83.125.13	TCP	66	51666 > 35364 [ACK]
137	16.989296	130.83.125.13	192.168.178.20	TCP	66	35364 > 51666 [FIN, ACK]
138	16.989415	192.168.178.20	130.83.125.13	TCP	66	51666 > 35364 [ACK]
139	16.990224	130.83.125.13	192.168.178.20	FTP	89	Response: 226 Transfer complete