L91B - Security Problems in TCP/IP

Security Problems

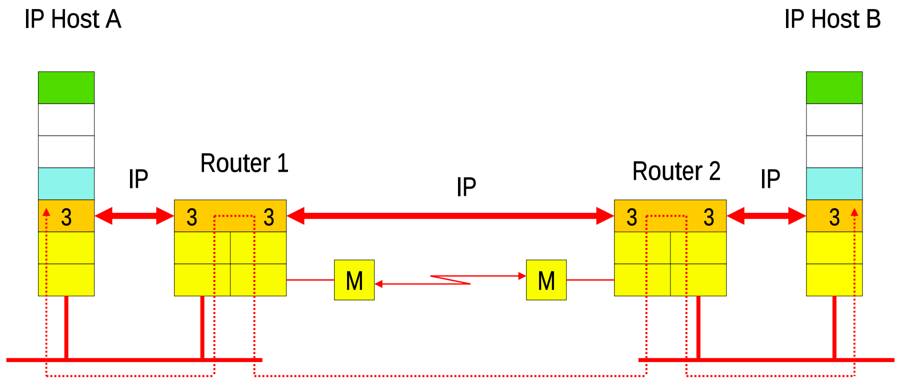
TCP/IP Level

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

IP and OSI Network Layer 3

Layer 3 Protocol = IP
Layer 3 Routing Protocols = RIP, OSPF, EIGRP, BGP


IP Host A

IP Host B

IP

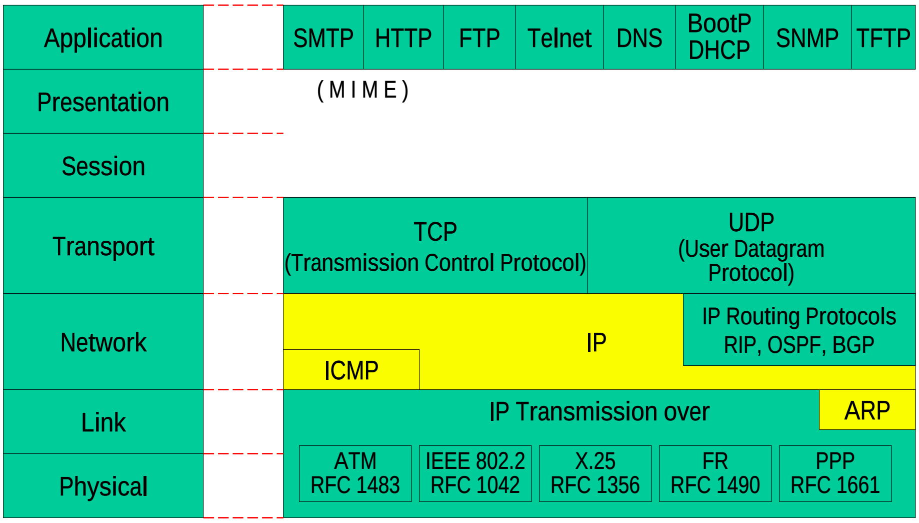
Router 1

IP

Router 2

IP

IP Related Protocols


Application

Presentation

Session

Transport

Network

Link

Physical

SMTP HTTP FTP Telnet DNS BootP DHCP SNMP TFTP

(M I M E)

TCP (Transmission Control Protocol)

UDP (User Datagram Protocol)

IP

IP Routing Protocols RIP, OSPF, BGP

ICMP

IP Transmission over

ARP

ATM RFC 1483

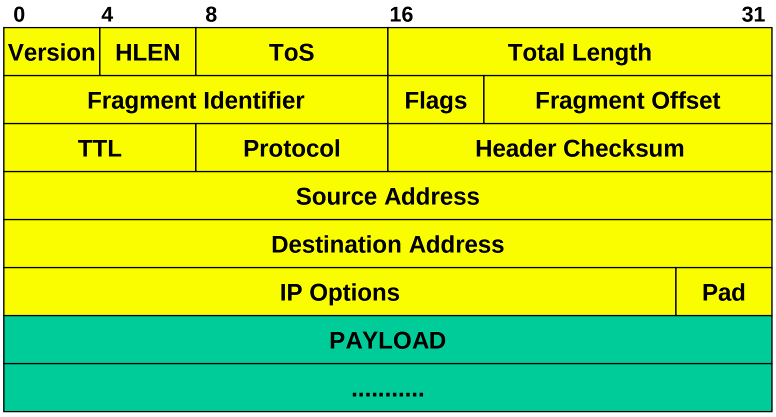
IEEE 802.2 RFC 1042

X.25 RFC 1356

FR RFC 1490

PPP RFC 1661

IP Header



0	4	8	16	31
Version	HLEN	ToS	Total Length	
Fragment Identifier			Flags	Fragment Offset
TTL		Protocol	Header Checksum	
Source Address				
Destination Address				
IP Options				Pad
PAYLOAD				
....				

IP Header Entries 1

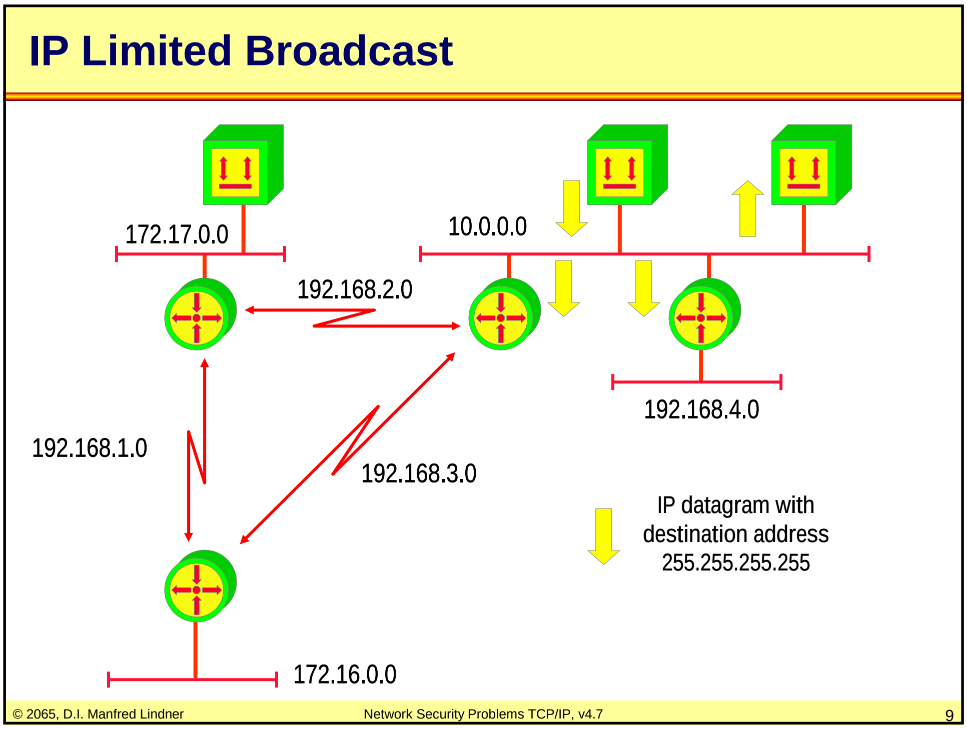
- **Version**
  - Version of the IP protocol
  - Current version is 4
  - Useful for testing or for migration to a new version, e.g. "IP next generation“ (IPv6)
- **HLEN**
  - Length of the header in 32 bit words
  - Different header lengths result from IP options
    - HLEN 5 to 15 = 20 to 60 octets

IP Header Entries 2

- **Total Length**
  - Total length of the IP datagram (header + data) in octets
  - If fragmented: length of fragment
  - Datagram size max. = 65535 octets
  - Each host has to accept datagram's of at least 576 octets
    - either as a complete datagram or for reassembly

IP Header Entries 3

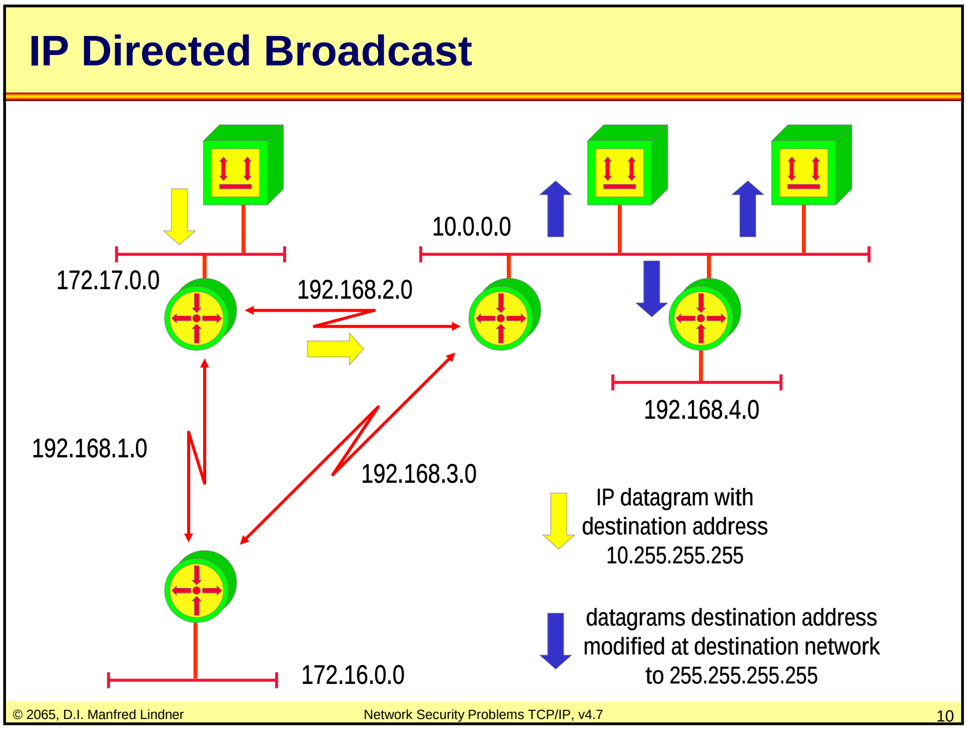
- **Protocol**
  - Indicates the higher layer protocols
    - Examples are: 1 (ICMP), 6 (TCP), 8 (EGP), 17 (UDP), 89 (OSPF) etc.
  - 100 different IP protocol types are registered so far
- **Source IP Address**
  - IP address of the source (sender) of a datagram
- **Destination IP Address**
  - IP address of the receiver (destination) of a datagram
- **Pad**
  - "0"-octets to fill the header to a 32 bit boundary



IP Limited Broadcast

172.17.0.0
10.0.0.0
192.168.2.0
192.168.4.0
192.168.1.0
192.168.3.0
172.16.0.0

IP datagram with
destination address
255.255.255.255



IP Directed Broadcast

10.0.0.0
172.17.0.0
192.168.2.0
192.168.4.0
192.168.1.0
192.168.3.0
172.16.0.0

IP datagram with
destination address
10.255.255.255

datagrams destination address
modified at destination network
to 255.255.255.255

IP Header Entries 4

- **TTL Time To Live**
  - Limits the lifetime of a datagram in the network (Units are seconds, range 0-255)
  - Is set by the source to a starting value. 32 to 64 are common values, the current recommended value is 64 (RFC1700)
  - Every router decrements the TTL by the processing/waiting time. If the time is less than one second, TTL is decremented by one ("TTL = hop count").
  - If TTL reaches 0, the datagram (fragment) is discarded.
  - An end system can use the remaining TTL value of the first arriving fragment to set the reassembly timer.

IP Header Entries 5

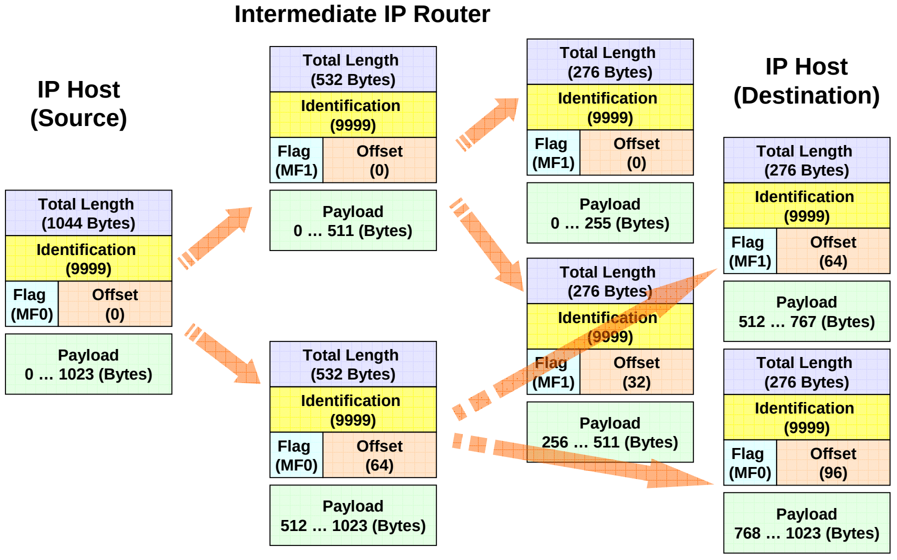
- **Identification (for fragmentation)**
  - Unique identification of a datagram, used for fragmentation and reassembly
  - In praxis a hidden sequence number although not used because of connectionless behavior of IP
- **Flags (for fragmentation).**
  - DF (don't fragment)
    - If set: fragmentation is not allowed
    - Datagram's must be discarded by router if MTU (maximum transmission unit) size of next link is too small
  - MF (more fragments)
    - If set: more fragments of the same original datagram will follow

"0"	DF	MF	Fragment Offset

IP Header Entries 6

- **Fragment Offset**
  - Indicates the position of a fragment relative to the beginning of the original datagram
  - Offset is measured in multiples of 8 octets (64 bits)
  - The first fragment and unfragmented packets have an offset of 0
  - Fragments (except the last) must be a multiple of 8 octets
  - Fragments with the same combination of source address / destination address / protocol / identification will be reassembled to the original datagram

IP Fragmentation



Intermediate IP Router

IP Host (Source)

Total Length (1044 Bytes)
Identification (9999)
Flag (MF0) Offset (0)
Payload 0 … 1023 (Bytes)

Total Length (532 Bytes)
Identification (9999)
Flag (MF1) Offset (0)
Payload 0 … 511 (Bytes)

Total Length (532 Bytes)
Identification (9999)
Flag (MF0) Offset (64)
Payload 512 … 1023 (Bytes)

Total Length (276 Bytes)
Identification (9999)
Flag (MF1) Offset (0)
Payload 0 … 255 (Bytes)

Total Length (276 Bytes)
Identification (9999)
Flag (MF1) Offset (32)
Payload 256 … 511 (Bytes)

IP Host (Destination)

Total Length (276 Bytes)
Identification (9999)
Flag (MF1) Offset (64)
Payload 512 … 767 (Bytes)

Total Length (276 Bytes)
Identification (9999)
Flag (MF0) Offset (96)
Payload 768 … 1023 (Bytes)

Reassembly

- Reassembly is done at the destination, because fragments can take different paths
- Buffer space has to be provided at the receiver
- Some fragments may not arrive (unreliable nature of IP)
- Measures must be taken to free buffers if a packet can't be reconstructed in a timely manner
- The first arriving fragment of an IP packet (with MF=1 and/or Offset <> 0) starts a reassembly timer
- If the timer expires before the packet was reconstructed, all fragments will be discarded and the buffer is set free
- The reassembly timer limits the lifetime of an incomplete datagram and allows better use of buffer resources.

IP Header Entries 7

- **TOS field (Type Of Service)**
- **Old Meaning (RFC 1349)**
  - Tells the priority of a datagram (precedence bits) and the preferred network characteristics (low delay, high throughput, high reliability, low monetary cost.)
  - Precedence bits:
    - Define the handling of a datagram within the router
    - e.g. priority within the input / output queues
  - D, T, R and C bits:
    - Can be used to take a path decision for routing if multiple paths with different characteristics exist to the destination
      - needs one routing table per characteristic
    - TOS bits may be ignored by routers but may never lead to discarding a packet if the preferred service can't be provided

TOS Field Old Meaning (RFC 1349)

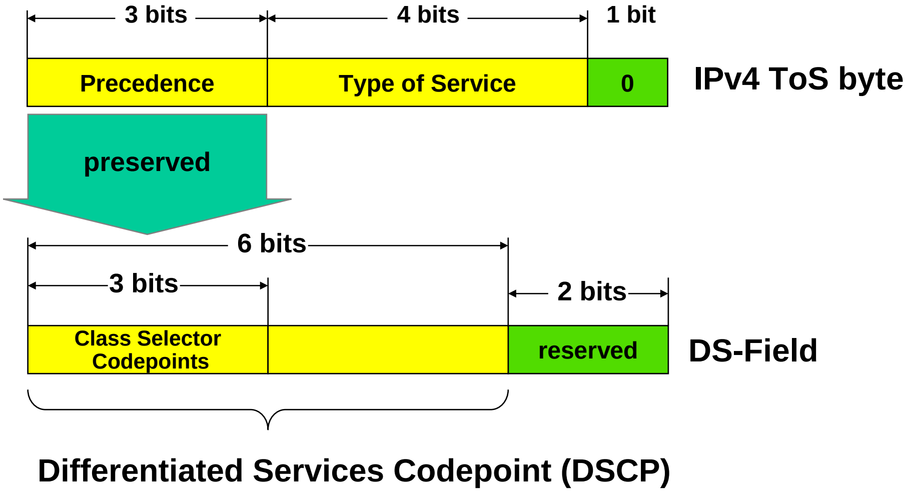
Precedence	D	T	R	C	"0"

Precedence (Priority):	DTRC bits:		
111 Network Control	0 0 0 0 . .	. . . .	normal service
110 Internetwork Control	1 0 0 0 D	Delay	min. delay
101 Critic/ECP	0 1 0 0 T	Throughput	max. throughput
100 Flash Override	0 0 1 0 R	Reliability	max. reliability
011 Flash	0 0 0 1 C	Cost	min. cost
010 Immediate			
001 Priority	No other values are defined but have to be accepted (ignored) by a router or host.		
000 Routine			

IPv4 TOS Recycling

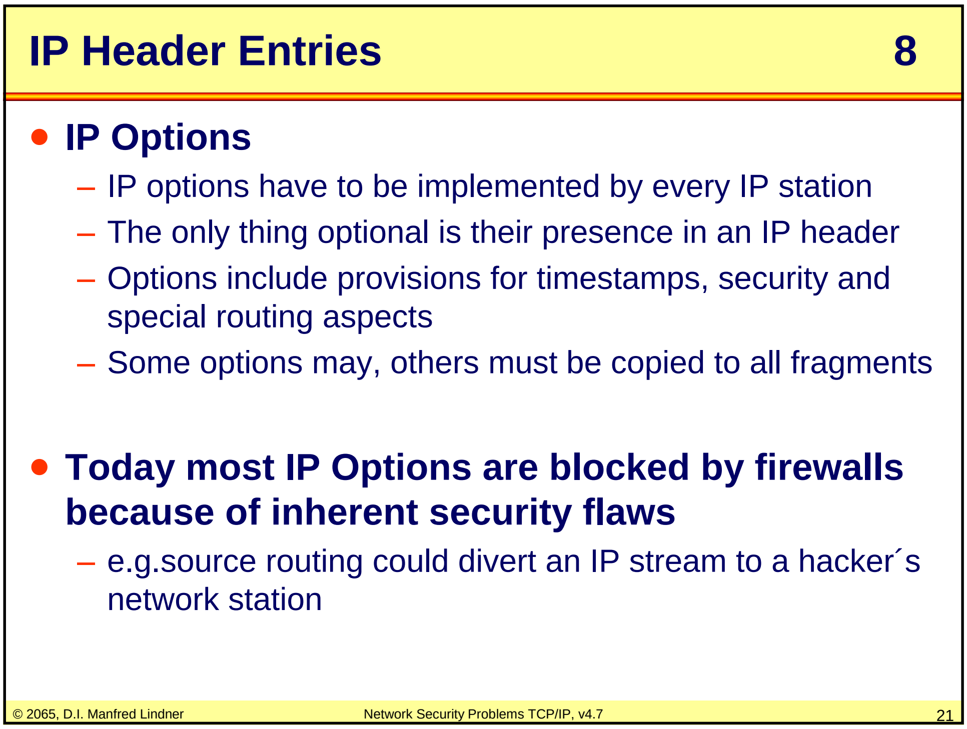
- **IPv4 TOS field was redefined by the IETF to become the "Differentiated Service CodePoint" (DSCP)**
- **Now the DSCP field is used to label the traffic class of a flow**
  - a flow is a given communication relationship (session) between two IP hosts
  - IP datagram's of a flow have the same
    - Source IP Address
    - Destination IP Address
    - Protocol Number
    - TCP/UDP Source Port
    - TCP/UDP Destination Port

IPv4 TOS Recycling


3 bits
4 bits
1 bit
Precedence
Type of Service
0
IPv4 ToS byte
preserved
6 bits
3 bits
2 bits
Class Selector Codepoints
reserved
DS-Field
Differentiated Services Codepoint (DSCP)

DSCP Usage

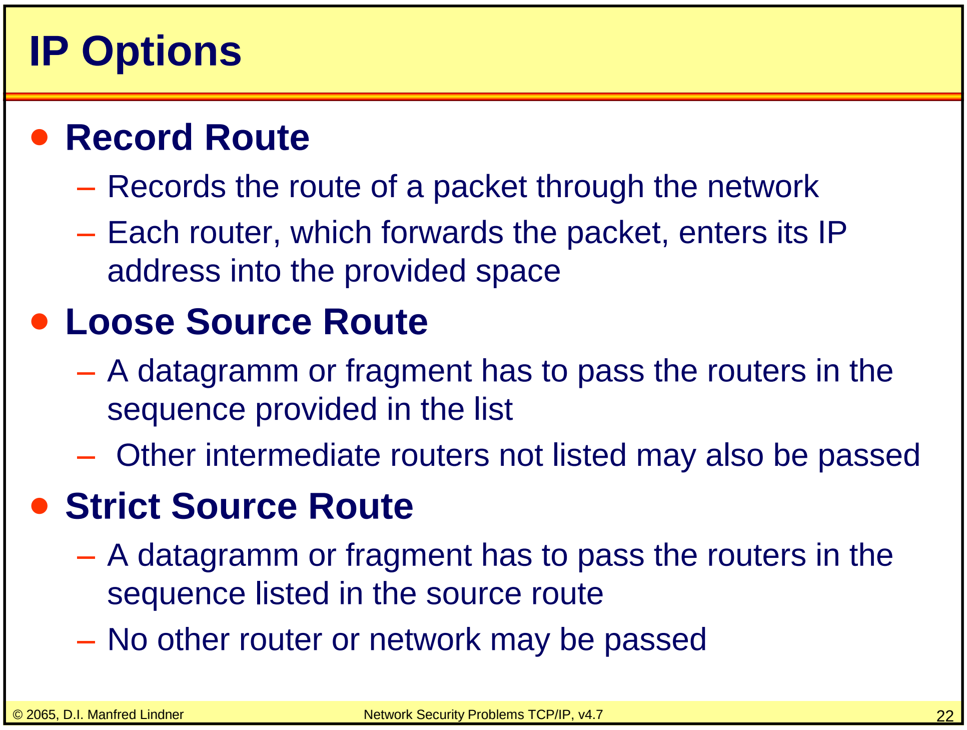
- **Important for IP QoS (Quality of Service)**
  - IP QoS Differentiated Services Model
    - RFC 2474: "Definition of the Differentiated Service Field in the IPv4 and IPv6 Headers"
    - RFC 2475: "An Architecture for Differentiated Services”
  - Remember
    - IP is basically a Best Effort Service, therefore not suited for interactive real-time traffic like voice and video
  - Using DSCP a IP datagram can be labelled at the border of IP QoS domain
    - with a certain traffic class
  - Traffic class will receive a defined handling within in IP QoS Domain
    - e.g. limited delay, guaranteed throughput



IP Header Entries

8

- **IP Options**
  - IP options have to be implemented by every IP station
  - The only thing optional is their presence in an IP header
  - Options include provisions for timestamps, security and special routing aspects
  - Some options may, others must be copied to all fragments
- **Today most IP Options are blocked by firewalls because of inherent security flaws**
  - e.g.source routing could divert an IP stream to a hacker´s network station



IP Options

- **Record Route**
  - Records the route of a packet through the network
  - Each router, which forwards the packet, enters its IP address into the provided space
- **Loose Source Route**
  - A datagramm or fragment has to pass the routers in the sequence provided in the list
  - Other intermediate routers not listed may also be passed
- **Strict Source Route**
  - A datagramm or fragment has to pass the routers in the sequence listed in the source route
  - No other router or network may be passed

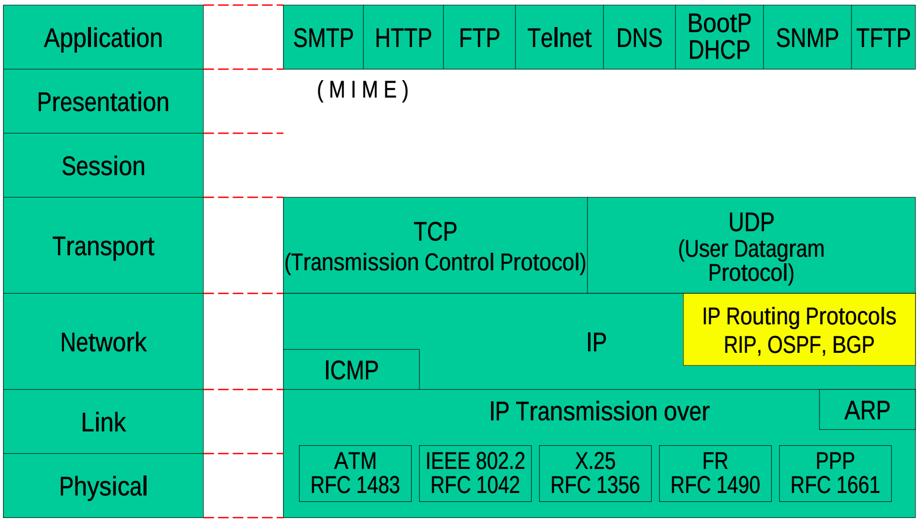
IP Routing

- **routing can be either**
  - static
    - routing tables are preconfigured by network administrator
    - non-responsive to topology changes
    - can be labor intensive to set up and modify in complex networks
    - but may be considered to be secure
  - or dynamic
    - routing tables are dynamically updated with information received from other routers
    - communication between routers is done by routing protocols
    - responsive to topology changes
    - routing messages may be faked by an intruder either to DoS or to redirect to the intruders machine
      - use authentication and integrity checking e.g. by keyed-MD5 signatures

Dynamic Routing

- **dynamic routing**
  - routing tables are dynamically updated with information from other routers done by routing protocols
  - routing protocol
    - discovers current network topology
    - determines the best path to every reachable network
    - stores information about best paths in the routing table
  - metric information is necessary for best path decision
    - in most cases summarization along the a given path of static preconfigured values
      - hops, interface cost, interface bandwidth, interface delay, etc.
  - two basic technologies
    - distance vector, link state

IP Routing Related Protocols



Application | Presentation | Session | Transport | Network | Link | Physical

SMTP | HTTP | FTP | Telnet | DNS | BootP DHCP | SNMP | TFTP

(MIME)

TCP (Transmission Control Protocol)

UDP (User Datagram Protocol)

IP

IP Routing Protocols RIP, OSPF, BGP

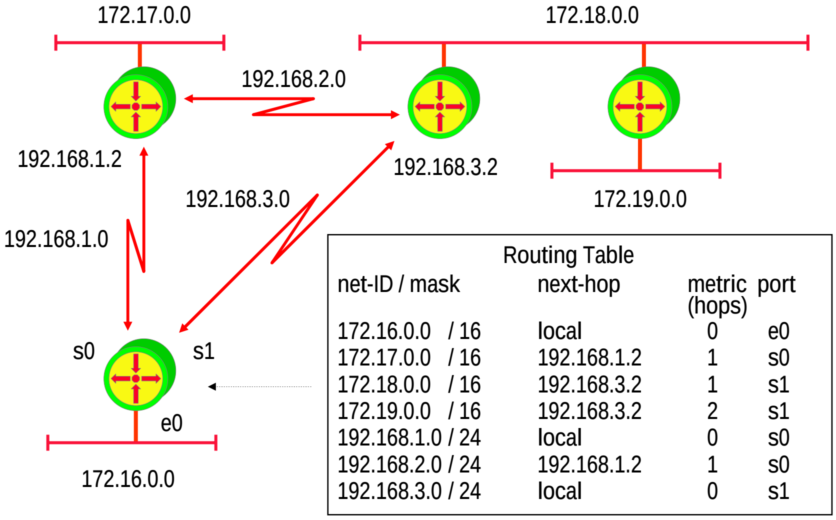
ICMP

IP Transmission over

ARP

ATM RFC 1483 | IEEE 802.2 RFC 1042 | X.25 RFC 1356 | FR RFC 1490 | PPP RFC 1661

Routing Table Example



Routing Table

net-ID / mask	next-hop	metric (hops)	port
172.16.0.0 / 16	local	0	e0
172.17.0.0 / 16	192.168.1.2	1	s0
172.18.0.0 / 16	192.168.3.2	1	s1
172.19.0.0 / 16	192.168.3.2	2	s1
192.168.1.0 / 24	local	0	s0
192.168.2.0 / 24	192.168.1.2	1	s0
192.168.3.0 / 24	local	0	s1

IP Routing Paradigm

- **Destination Based Routing**
  - source address is not taken into account for the forward decision
- **Hop by Hop Routing**
  - IP datagram's follow the path, which is pointed by the current state of the routing tables
- **Least Cost Routing**
  - normally only the best path is considered for forwarding of IP datagram's
  - alternate paths will not be used in order to reach a given destination
    - note:some methods allow load balancing if paths are equal

ICMP (Internet Control Message Protocol)

- **The datagram service of IP doesn't guarantee or acknowledge the delivery of a datagram**
- **ICMP generates error messages to enhance the reliability and to provide information about errors and packet loss in the network**
- **ICMP allows to request information for debugging and diagnosis**
- **ICMP has to be supported by every IP station but different implementation may vary in the way how ICMP messages are responded to**

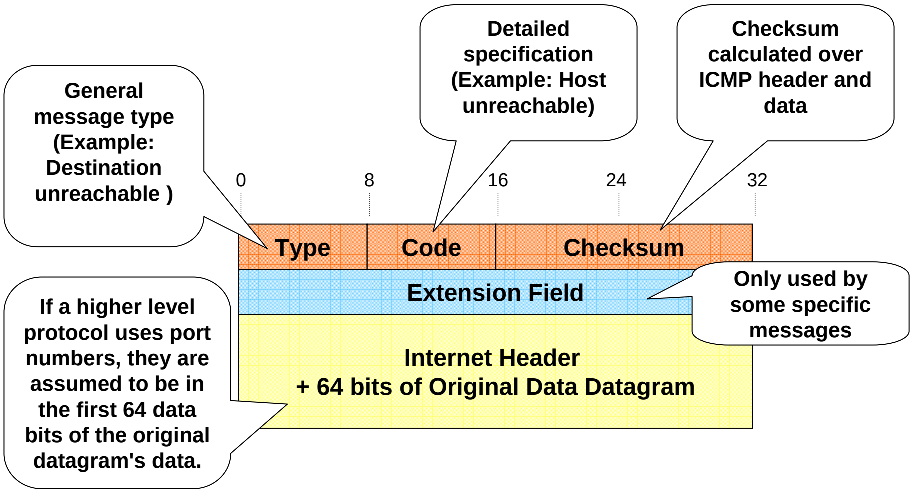
ICMP (RFC 792)

- **Principles of operation:**
  - The IP station (router or destination), which detects any transmission problems, generates an ICMP message
  - The ICMP message is addressed to the originating station (sender of the original IP packet)
- **ICMP messages are sent as IP packets**
  - protocol field = 1, ICMP header and code in the IP data area
- **Analysis of ICMP messages**
  - through network management systems or statistic programs can give valuable hints for network administrators

Important Rule

- **If a IP datagram carrying an ICMP message cannot be delivered**
  - No additional ICMP error message is generated to avoid an ICMP avalanche
  - "ICMP must not invoke ICMP"
- **Exception: PING command**
  - Echo request and echo response

ICMP Message Format



General message type (Example: Destination unreachable)

Detailed specification (Example: Host unreachable)

Checksum calculated over ICMP header and data

Only used by some specific messages

If a higher level protocol uses port numbers, they are assumed to be in the first 64 data bits of the original datagram's data.

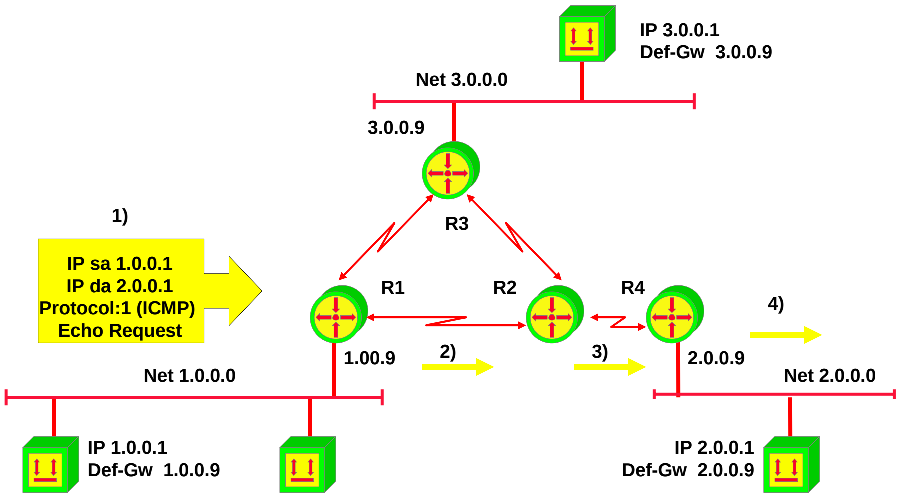
Type Field

0	Echo reply ("Ping")
3	Destination Unreachable Reason specified in Code
4	Source Quench (decrease data rate of sender) Theoretical Flow Control Possibility of IP
5	Redirect (use different router) More information in Code
8	Echo Request ("PING")
11	Time Exceeded (code = 0 time to live exceeded in transit code = 1 reassembly timer expired)
12	Parameter Problem (IP header)
13/14	Time Stamp Request / Time Stamp Reply
15/16	Information Request/ Reply (finding the Net-ID of the network; e.g. SLIP)
17/18	Address Mask Request / Reply

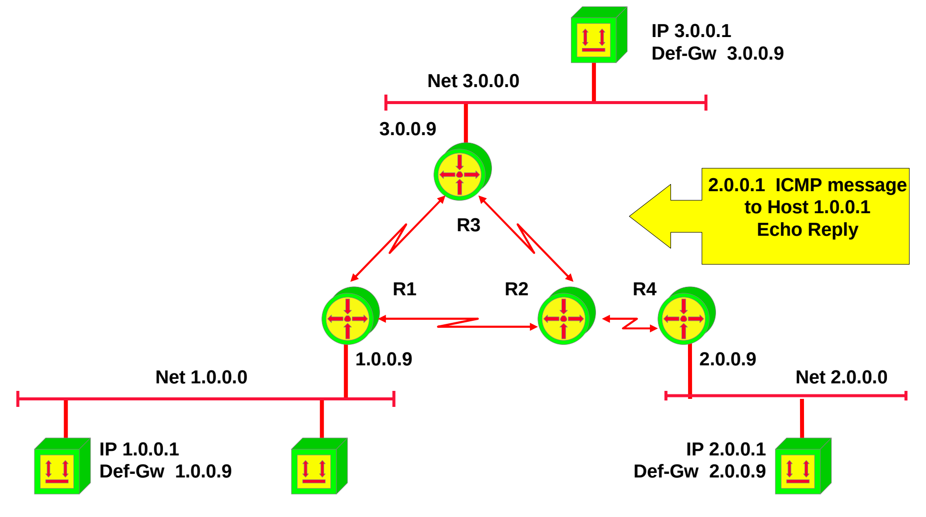
Using ICMP Types

0, 8	"PING" testing whether an IP station (router or end system) can be reached and is operational
3, 11, 12	Signaling errors concerning reachability, TTL / reassambly timeouts and errors in the IP header
4	Flow control (only possibility to signal a possible buffer overflow)
5	Signaling of alternative (shorter) routes to a target
13 - 18	Diagnosis or management

Ping 1.0.0.1 - > 2.0.0.1

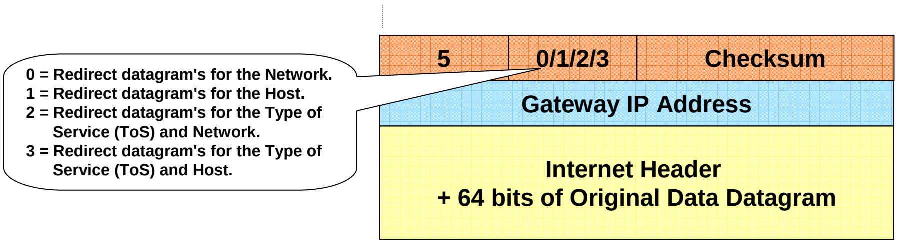

IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
3.0.0.9
R3
1)
IP sa 1.0.0.1
IP da 2.0.0.1
Protocol:1 (ICMP)
Echo Request
R1
R2
R4
4)
1.00.9
2)
3)
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

Ping Echo 2.0.0.1 - > 1.0.0.1


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
3.0.0.9
R3
2.0.0.1 ICMP message
to Host 1.0.0.1
Echo Reply
R1
R2
R4
1.0.0.9
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

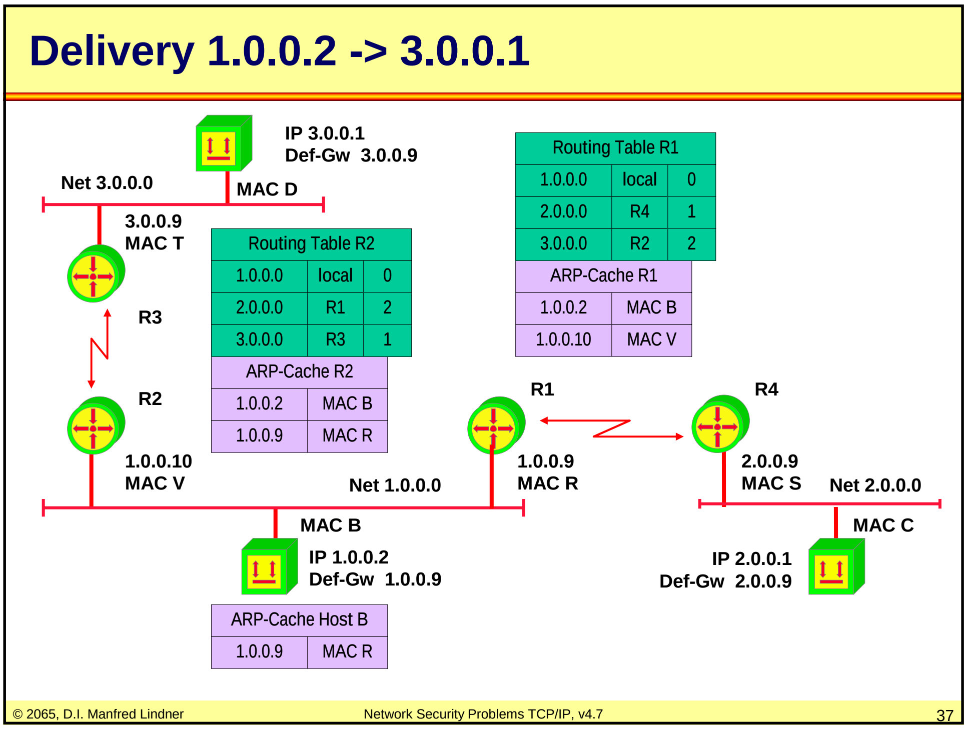
Code Field for ICMP Type 5 (Redirect)

- If a router knows of a better (faster, shorter) path to a target then it will notify the sender through ICMP redirect
  - In any case the router will still forward the packets on the inefficient path
  - Datagram's will be sent twice through a LAN, if the sender ignores the redirect message



0 = Redirect datagram's for the Network.
1 = Redirect datagram's for the Host.
2 = Redirect datagram's for the Type of Service (ToS) and Network.
3 = Redirect datagram's for the Type of Service (ToS) and Host.

5	0/1/2/3	Checksum
Gateway IP Address		
Internet Header + 64 bits of Original Data Datagram		



Delivery 1.0.0.2 -> 3.0.0.1

IP 3.0.0.1
Def-Gw 3.0.0.9

Net 3.0.0.0

MAC D

3.0.0.9
MAC T

R3

R2

1.0.0.10
MAC V

Routing Table R2		
1.0.0.0	local	0
2.0.0.0	R1	2
3.0.0.0	R3	1

ARP-Cache R2	
1.0.0.2	MAC B
1.0.0.9	MAC R

Routing Table R1		
1.0.0.0	local	0
2.0.0.0	R4	1
3.0.0.0	R2	2

ARP-Cache R1	
1.0.0.2	MAC B
1.0.0.10	MAC V

R1

R4

1.0.0.9
MAC R

2.0.0.9
MAC S

Net 1.0.0.0

Net 2.0.0.0

MAC B

IP 1.0.0.2
Def-Gw 1.0.0.9

MAC C

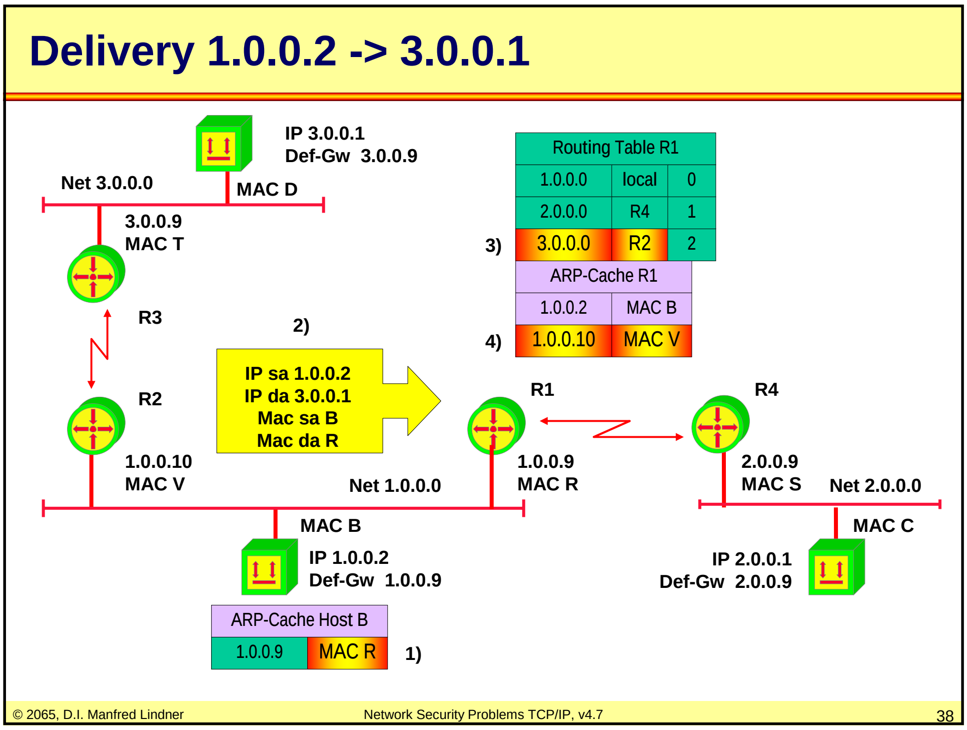
IP 2.0.0.1
Def-Gw 2.0.0.9

ARP-Cache Host B	
1.0.0.9	MAC R

© 2065, D.I. Manfred Lindner

Network Security Problems TCP/IP, v4.7

37



Delivery 1.0.0.2 -> 3.0.0.1

IP 3.0.0.1
Def-Gw 3.0.0.9

Net 3.0.0.0

MAC D

3.0.0.9
MAC T

R3

R2

1.0.0.10
MAC V

2)

IP sa 1.0.0.2
IP da 3.0.0.1
Mac sa B
Mac da R

3)

Routing Table R1		
1.0.0.0	local	0
2.0.0.0	R4	1
3.0.0.0	R2	2

4)

ARP-Cache R1	
1.0.0.2	MAC B
1.0.0.10	MAC V

R1

R4

1.0.0.9
MAC R

2.0.0.9
MAC S

Net 1.0.0.0

Net 2.0.0.0

MAC B

IP 1.0.0.2
Def-Gw 1.0.0.9

MAC C

IP 2.0.0.1
Def-Gw 2.0.0.9

ARP-Cache Host B	
1.0.0.9	MAC R

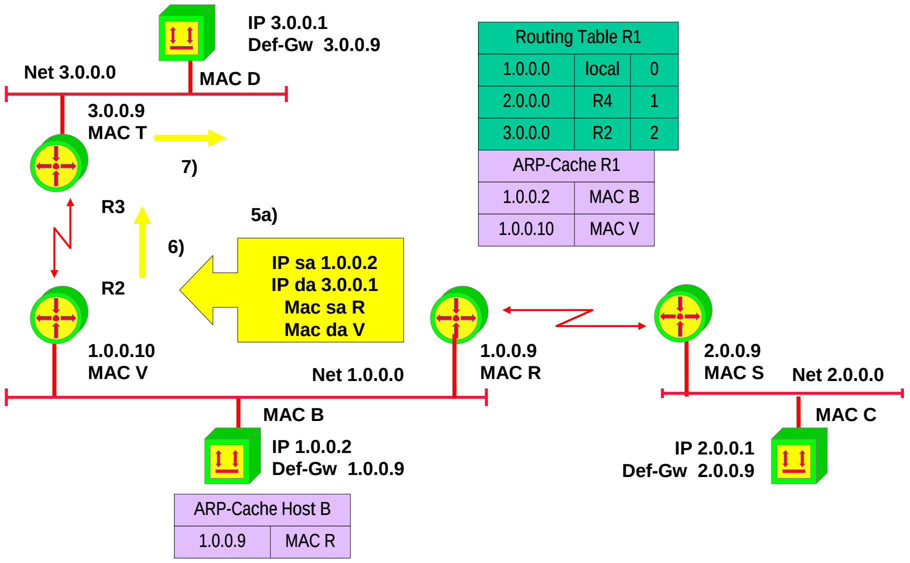
1)

© 2065, D.I. Manfred Lindner

Network Security Problems TCP/IP, v4.7

38

Delivery 1.0.0.2 -> 3.0.0.1


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
7)
R3
6)
5a)
R2
1.0.0.10
MAC V
Net 1.0.0.0
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 2.0.0.0
MAC C
MAC B
IP 1.0.0.2
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

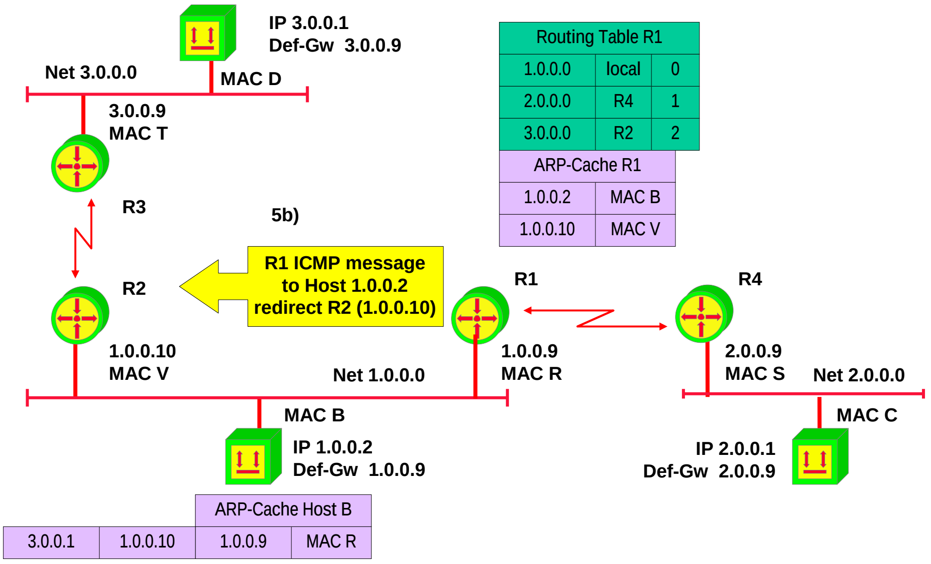
IP sa 1.0.0.2
IP da 3.0.0.1
Mac sa R
Mac da V

Routing Table R1		
1.0.0.0	local	0
2.0.0.0	R4	1
3.0.0.0	R2	2

ARP-Cache R1	
1.0.0.2	MAC B
1.0.0.10	MAC V

ARP-Cache Host B	
1.0.0.9	MAC R

ICMP redirect


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
5b)
R2
R1
R4
1.0.0.10
MAC V
Net 1.0.0.0
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 2.0.0.0
MAC C
MAC B
IP 1.0.0.2
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

R1 ICMP message
to Host 1.0.0.2
redirect R2 (1.0.0.10)

Routing Table R1		
1.0.0.0	local	0
2.0.0.0	R4	1
3.0.0.0	R2	2

ARP-Cache R1	
1.0.0.2	MAC B
1.0.0.10	MAC V

		ARP-Cache Host B	
3.0.0.1	1.0.0.10	1.0.0.9	MAC R

Code Field for ICMP Type 3 (destination unreachable)

0 ... Network unreachable: no path to network known or network down; generated by intermediate or far-end router
1 ... Host unreachable: Host-ID can't be resolved or host not responding; generated by far-end router
2 ... Protocol unreachable: protocol specified in IP header not available; generated by end system
3 ... Port unreachable: port (service) specified in layer 4 not available; generated by end system
4 ... Fragmentation needed and do not fragment bit set: DF bit =1 but the packet is too big for the network (MTU); generated by router
5 ... Source route failed: Path in IP Options couldn't be followed; generated by intermediate or far-end router

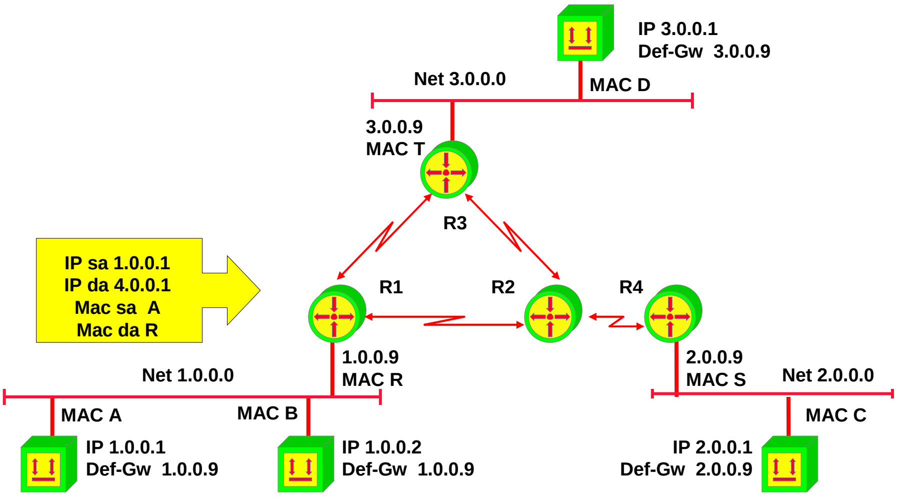
Code Field for Type 3 (destination unreachable)

See RFC1122 (Host Requirements) page 38:

The following additional codes are hereby defined:

6 … destination network unknown
7 … destination host unknown
8 … source host isolated
9 … communication with destination network administratively prohibited
10 … communication with destination host administratively prohibited
11 … network unreachable for type of service
12 … host unreachable for type of service

Delivery 1.0.0.1 - > 4.0.0.1



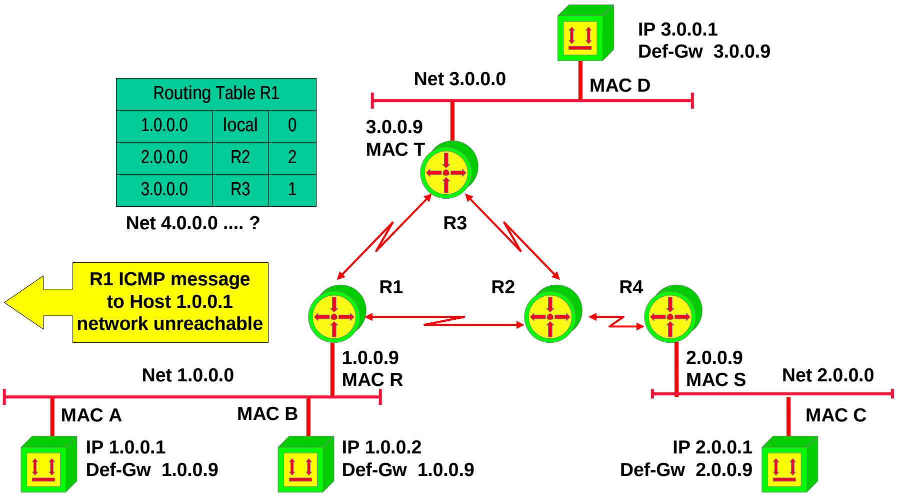
IP sa 1.0.0.1
IP da 4.0.0.1
Mac sa A
Mac da R

IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
R1
R2
R4
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 1.0.0.0
Net 2.0.0.0
MAC A
MAC B
MAC C
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 1.0.0.2
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

ICMP network unreachable

Routing Table R1		
1.0.0.0	local	0
2.0.0.0	R2	2
3.0.0.0	R3	1

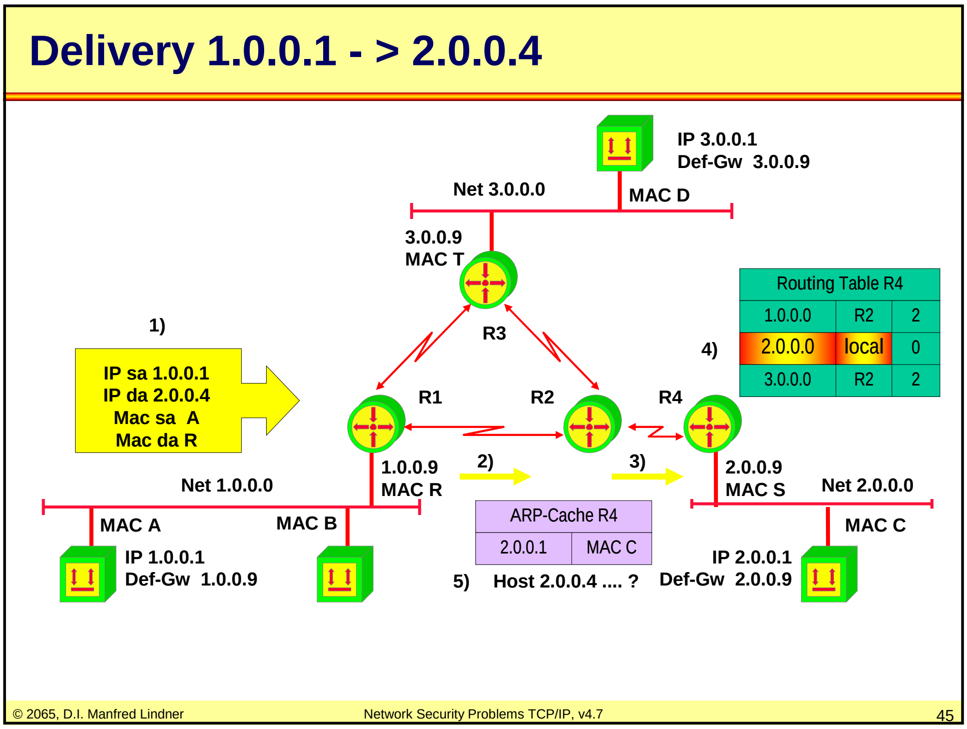
Net 4.0.0.0 .... ?



R1 ICMP message
to Host 1.0.0.1
network unreachable

IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
R1
R2
R4
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 1.0.0.0
Net 2.0.0.0
MAC A
MAC B
MAC C
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 1.0.0.2
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

Delivery 1.0.0.1 - > 2.0.0.4


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
R1
R2
R4
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 1.0.0.0
Net 2.0.0.0
MAC A
MAC B
MAC C
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9
2)
3)
4)

1)

IP sa 1.0.0.1
IP da 2.0.0.4
Mac sa A
Mac da R

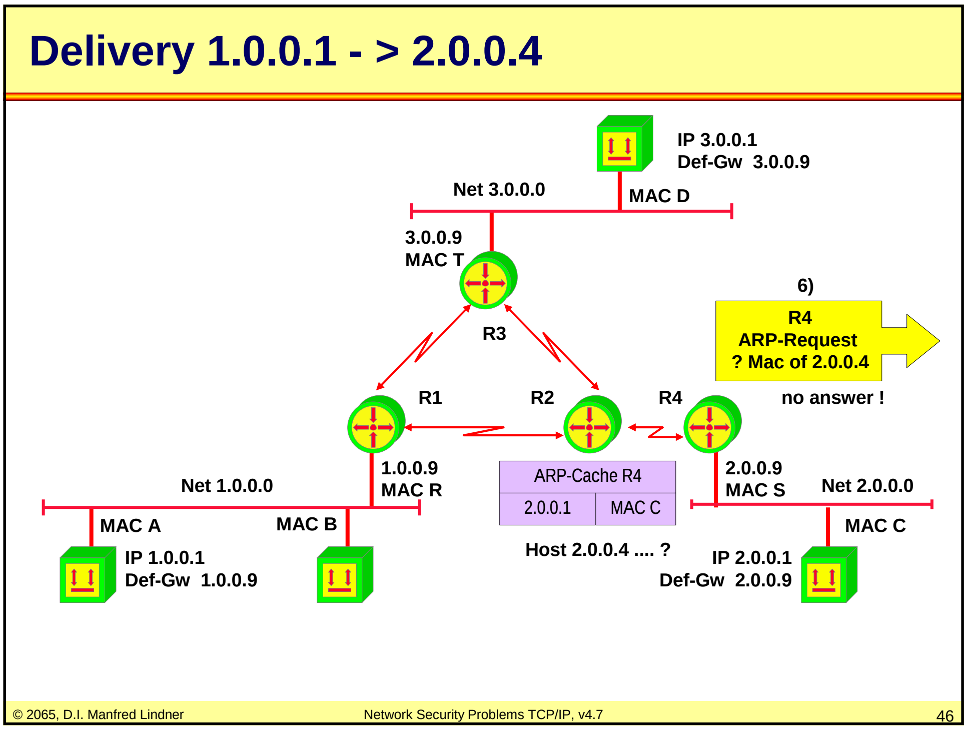
Routing Table R4		
1.0.0.0	R2	2
2.0.0.0	local	0
3.0.0.0	R2	2

ARP-Cache R4	
2.0.0.1	MAC C

5) **Host 2.0.0.4 .... ?**

© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 45

Delivery 1.0.0.1 - > 2.0.0.4


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
R1
R2
R4
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 1.0.0.0
Net 2.0.0.0
MAC A
MAC B
MAC C
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

6)

R4
ARP-Request
? Mac of 2.0.0.4

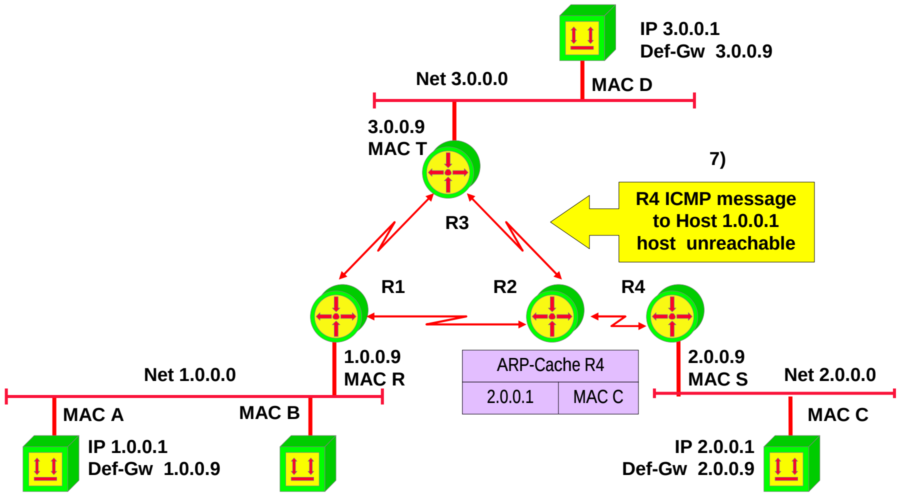
no answer !

ARP-Cache R4	
2.0.0.1	MAC C

Host 2.0.0.4 .... ?

© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 46

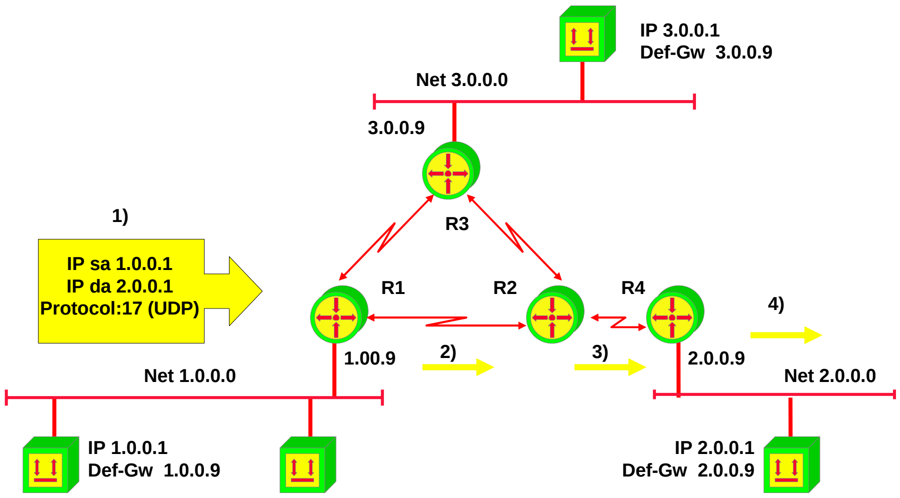
ICMP host unreachable


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
MAC D
3.0.0.9
MAC T
R3
7)
R1
R2
R4
1.0.0.9
MAC R
2.0.0.9
MAC S
Net 1.0.0.0
Net 2.0.0.0
MAC A
MAC B
MAC C
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

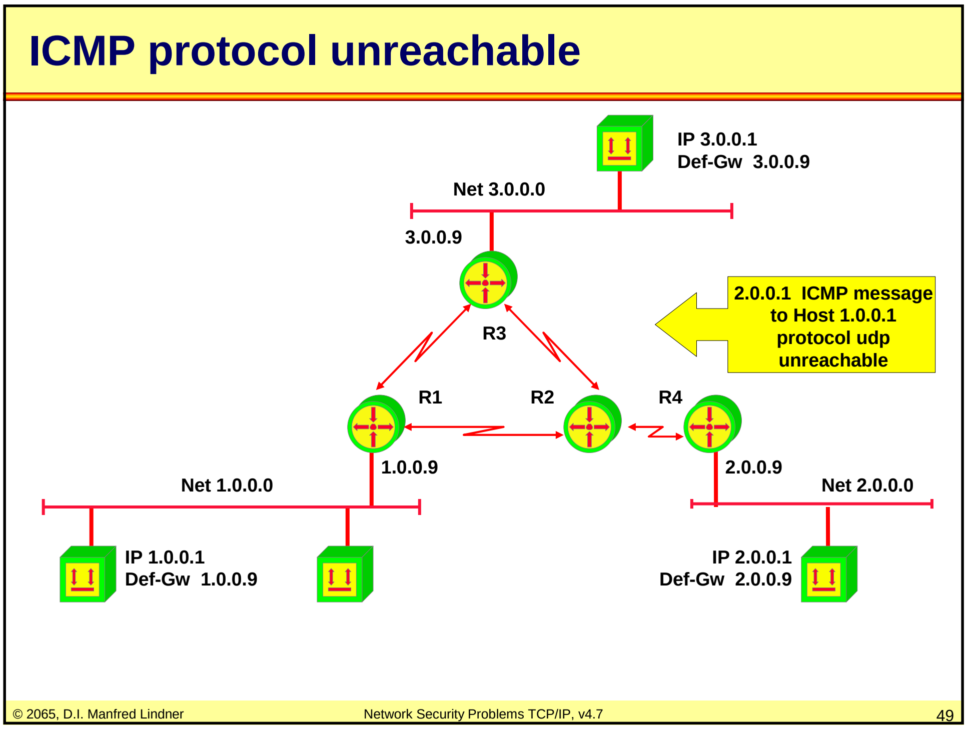
R4 ICMP message
to Host 1.0.0.1
host unreachable

ARP-Cache R4	
2.0.0.1	MAC C

Delivery 1.0.0.1 - > 2.0.0.1 (protocol udp)


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
3.0.0.9
R3
1)
R1
R2
R4
4)
1.00.9
2)
3)
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

IP sa 1.0.0.1
IP da 2.0.0.1
Protocol:17 (UDP)



ICMP protocol unreachable

IP 3.0.0.1
Def-Gw 3.0.0.9

Net 3.0.0.0

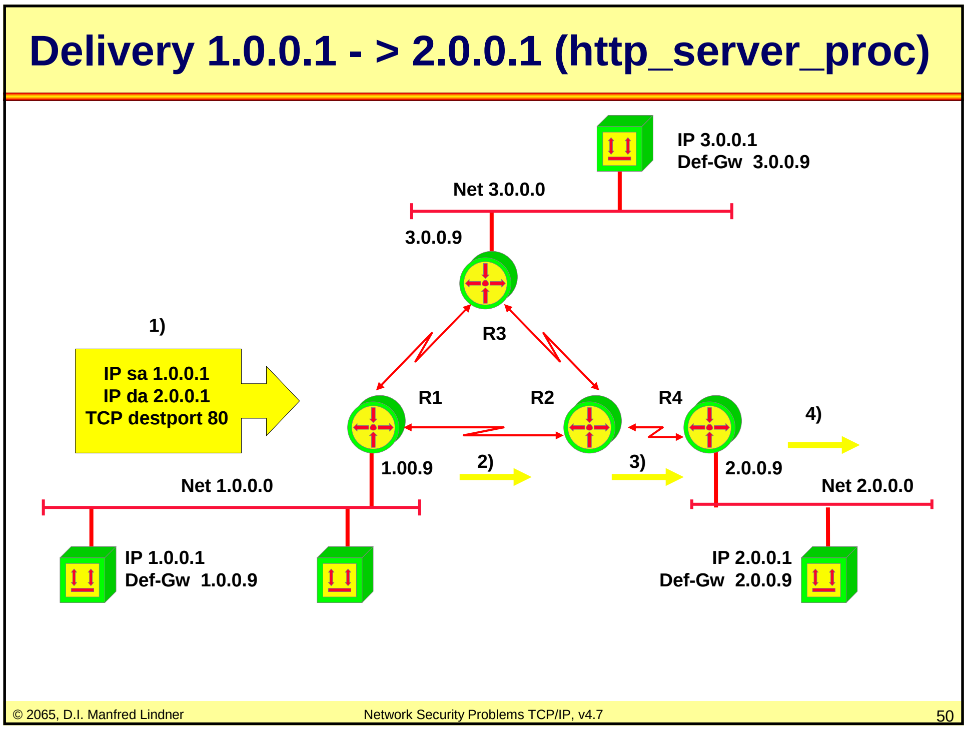
3.0.0.9

2.0.0.1 ICMP message
to Host 1.0.0.1
protocol udp
unreachable

R3
R1
R2
R4
1.0.0.9
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0

IP 1.0.0.1
Def-Gw 1.0.0.9

IP 2.0.0.1
Def-Gw 2.0.0.9



Delivery 1.0.0.1 - > 2.0.0.1 (http_server_proc)

IP 3.0.0.1
Def-Gw 3.0.0.9

Net 3.0.0.0

3.0.0.9

1)

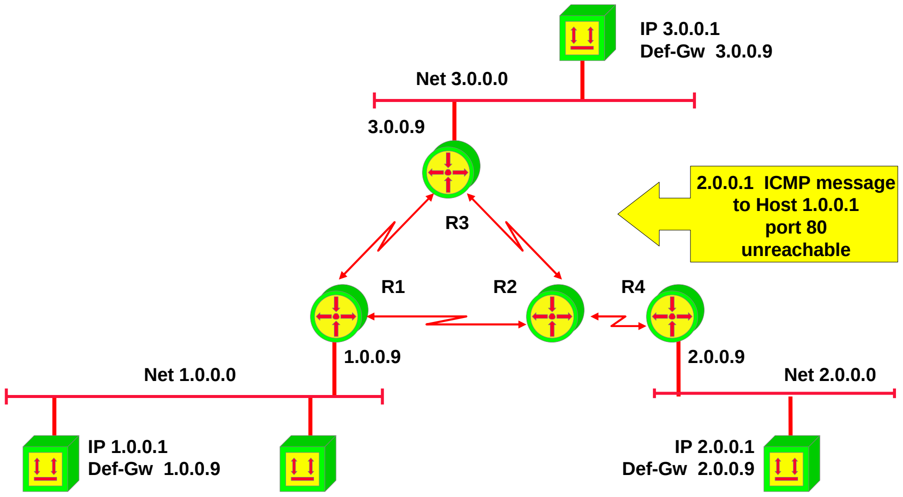
IP sa 1.0.0.1
IP da 2.0.0.1
TCP destport 80

R3
R1
R2
R4
4)
1.00.9
2)
3)
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0

IP 1.0.0.1
Def-Gw 1.0.0.9

IP 2.0.0.1
Def-Gw 2.0.0.9

ICMP port unreachable (no http_server_proc)


IP 3.0.0.1
Def-Gw 3.0.0.9
Net 3.0.0.0
3.0.0.9
R3
R1
R2
R4
1.0.0.9
2.0.0.9
Net 1.0.0.0
Net 2.0.0.0
IP 1.0.0.1
Def-Gw 1.0.0.9
IP 2.0.0.1
Def-Gw 2.0.0.9

2.0.0.1 ICMP message to Host 1.0.0.1 port 80 unreachable

Agenda

- IP
  - Review IP, ICMP
  - L3 Attacks on IP
- TCP
  - Review TCP
  - L3/L4 Attacks on TCP
- UDP
  - Review UDP
  - L3/L4 Attacks on UDP
- DNS
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- FTP
  - Review FTP
  - FTP Bounce Attack

Main Network Security Problems with IP 1

- L2 redirection to an intruder machine
  - ARP spoofing
  - Impersonate another IP station by faking the stations MAC address with the own MAC address in a foreign ARP cache
  - “Man-in-the-middle” attack with sniffing/manipulating of messages
- L3 redirection to an intruder machine
  - “Man-in-the-middle” attack with sniffing/manipulating of messages
- IP spoofing
  - Impersonate another IP station by using the stations IP address as source address in own packets (so called faked/forged address)
  - Used either for DoS attack or to break into a system which has authentication based on IP address
- DoS (Denial of Service)
  - Disturbing a machine which offers a service in the Internet
  - Often combined with IP spoofing

Main Network Security Problems with IP 2

- **Reconnaissance**
  - is the unauthorized discovery and mapping of systems, services, or vulnerabilities
  - is also known as information gathering, and in most cases, precedes an actual access or denial of service (DoS) attack
    - First, the malicious intruder typically conducts a **“Ping Sweep”** of the target network to determine which IP addresses are alive
    - Then the intruder determines which services or ports are active on the live IP addresses **(“Port Scan”)**
    - From this information, the intruder queries the ports to determine the type and version of the application and operating system running on the target host **(“OS Fingerprinting”)**

Reconnaissance Tools

- **May consist of**
  - Packet sniffer
    - Monitoring LAN traffic on a shared connection
      - Collision domain of a wired infrastructure
      - Wireless LANs as new upcoming challenge
  - Ping sweeps
    - ICMP echo request are separately sent to all IP addresses of an IP subnet
    - ICMP echo reply from the hosts which are alive
  - Port scan
    - Messages are sent to all ports (TCP, UDP) of a host to discover which services are running
  - Internet information queries
    - DNS, RIPE WHOIS, etc.

Attacks on IP 1

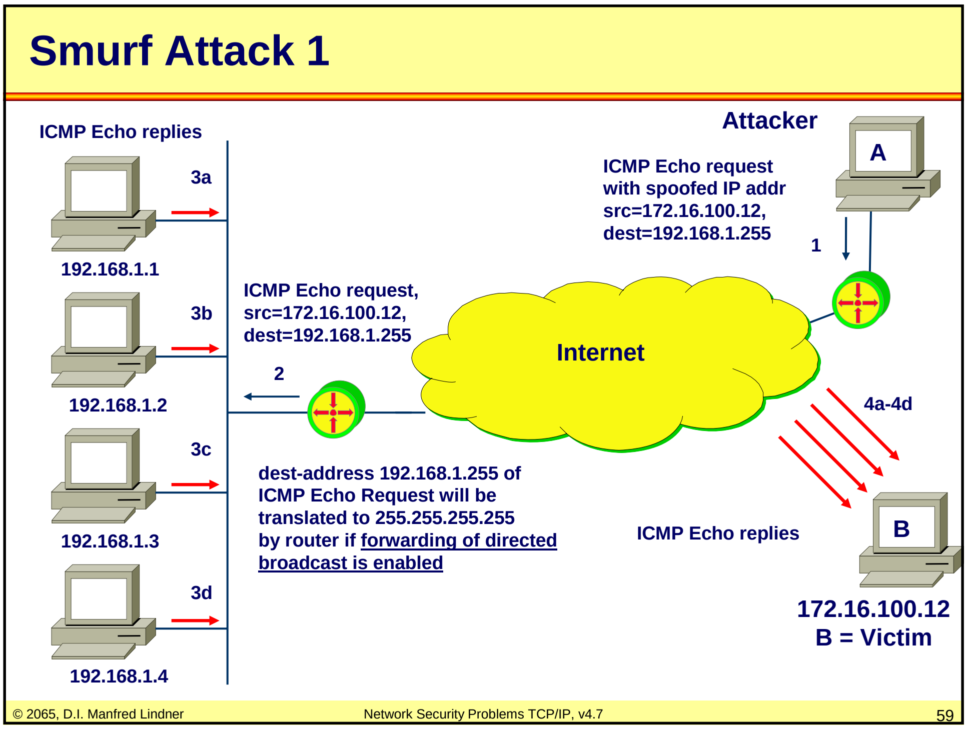
- **IP Fragmentation Attack**
  - “Ping of death”
  - maximum length of an IP packet = 65535
  - Sent a fragmented IP ping with a resulting length greater than 65,535 octets after reassembly
    - the offset of the last segment is such that the total size of the reassembled datagram is bigger than the maximum allowed size
    - kernel buffer overflow may cause a collapse of the OS
  - Type: DoS

Attacks on IP 2

- **IP Source Route Attack**
  - manipulating the IP Source Route Option
  - Type: Redirection
- **ICMP Redirect Attack**
  - manipulating ICMP redirect message
  - attacker sends a spoofed ICMP redirect message that appears to come maybe from the host‘s default gateway
  - Type: Redirection
- **Routing System Attacks**
  - manipulating routing messages of dynamic routing protocols like RIP, OSPF, BGP, …
  - Type: Redirection

Attacks on IP 3

- **ICMP Echo Attacks**
  - Figure out which hosts are active on a subnet
    - ICMP echo datagram's (“PING”) are sent to all hosts in a subnet
    - Attacker collects the replies and determines which hosts are alive
  - Smurf Attack
    - Intruder sends IP spoofed ICMP Echo Requests to subnets
    - Victim will get ICMP Echo Replies from every host of this subnet
    - Will work because of destination based routing behavior (router does not look to source address when forwarding a packet)
      - default on an Internet backbone router because of performance
      - should be changed with filter-list based on source address on egress (and ingress) router
      - egress router will become a so called packet level firewall
    - Type: DoS



Smurf Attack 1

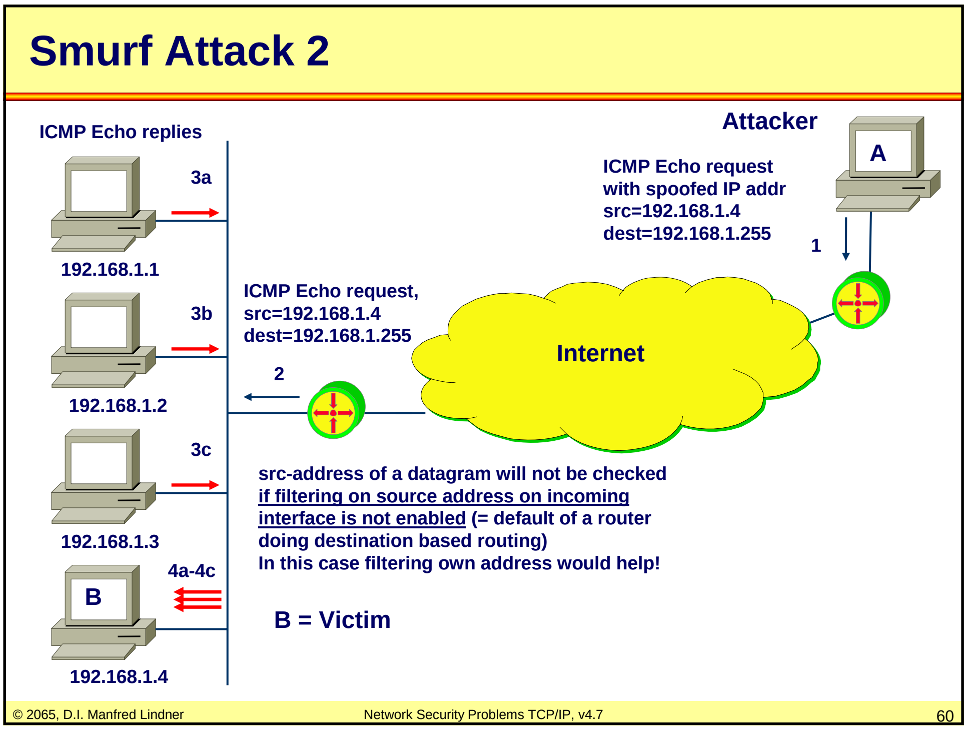
ICMP Echo replies
3a
3b
3c
3d
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
2
Internet
Attacker
A
1
4a-4d
ICMP Echo replies
B

ICMP Echo request with spoofed IP addr src=172.16.100.12, dest=192.168.1.255

ICMP Echo request, src=172.16.100.12, dest=192.168.1.255

dest-address 192.168.1.255 of ICMP Echo Request will be translated to 255.255.255.255 by router if forwarding of directed broadcast is enabled

172.16.100.12
B = Victim



Smurf Attack 2

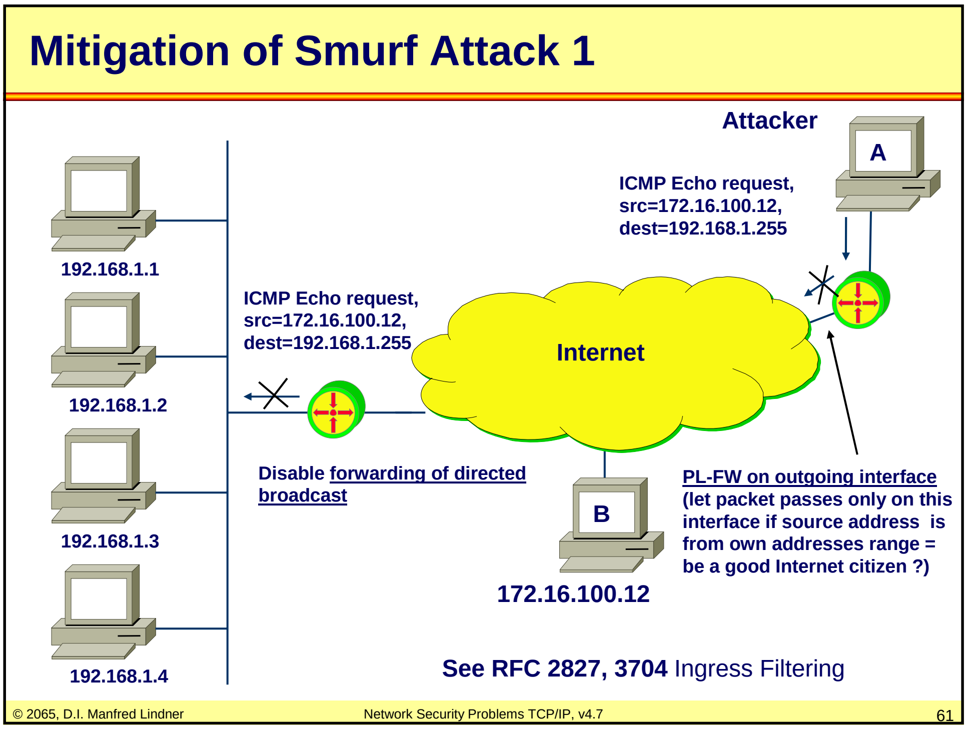
ICMP Echo replies
3a
3b
3c
4a-4c
192.168.1.1
192.168.1.2
192.168.1.3
B
192.168.1.4
2
Internet
Attacker
A
1

ICMP Echo request with spoofed IP addr src=192.168.1.4 dest=192.168.1.255

ICMP Echo request, src=192.168.1.4 dest=192.168.1.255

src-address of a datagram will not be checked if filtering on source address on incoming interface is not enabled (= default of a router doing destination based routing)
In this case filtering own address would help!

B = Victim



Mitigation of Smurf Attack 1

Attacker
A
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.4
Internet
B
172.16.100.12

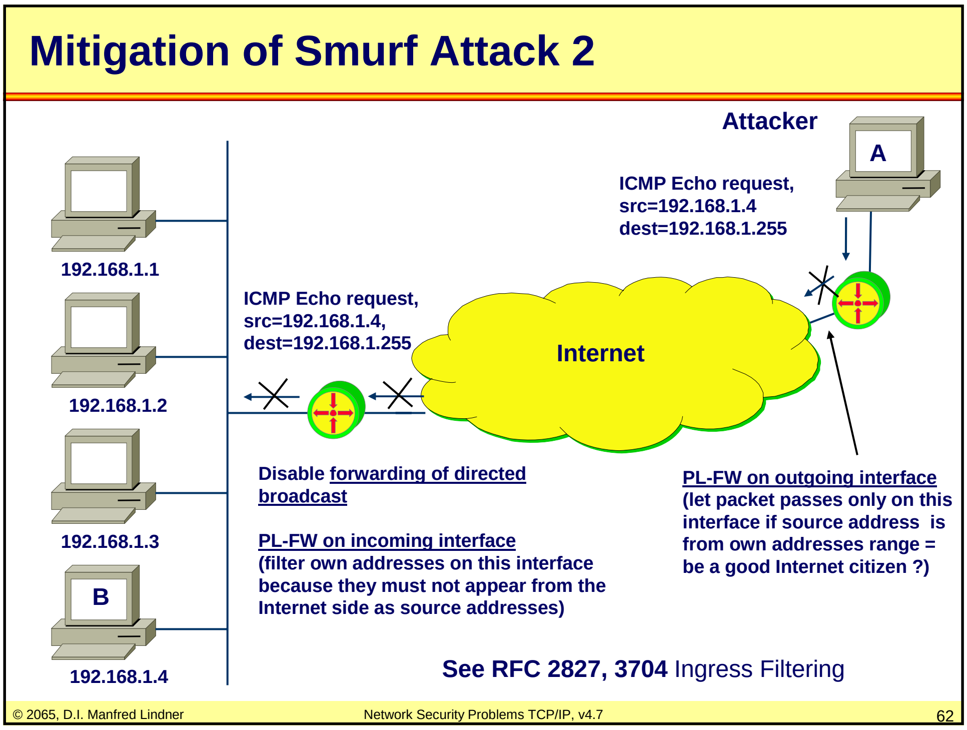
ICMP Echo request,
src=172.16.100.12,
dest=192.168.1.255

ICMP Echo request,
src=172.16.100.12,
dest=192.168.1.255

Disable forwarding of directed broadcast

PL-FW on outgoing interface
(let packet passes only on this interface if source address is from own addresses range = be a good Internet citizen ?)

See RFC 2827, 3704 Ingress Filtering



Mitigation of Smurf Attack 2

Attacker
A
192.168.1.1
192.168.1.2
192.168.1.3
B
192.168.1.4
Internet

ICMP Echo request,
src=192.168.1.4
dest=192.168.1.255

ICMP Echo request,
src=192.168.1.4,
dest=192.168.1.255

Disable forwarding of directed broadcast

PL-FW on incoming interface
(filter own addresses on this interface because they must not appear from the Internet side as source addresses)

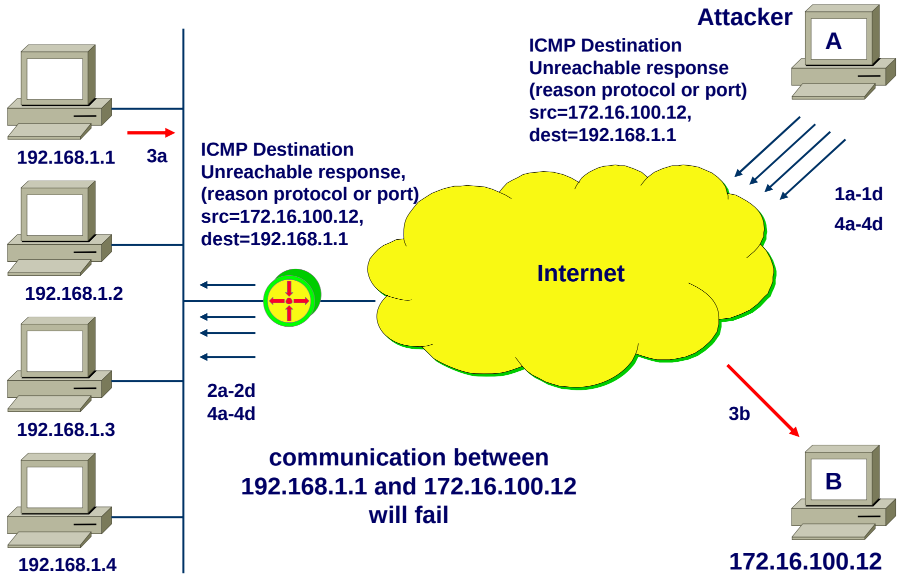
PL-FW on outgoing interface
(let packet passes only on this interface if source address is from own addresses range = be a good Internet citizen ?)

See RFC 2827, 3704 Ingress Filtering

Attacks on IP 4

- **ICMP Destination Unreachable Attacks**
  - Used to cut out nodes from the network
    - Attacker injects many forged destination unreachable messages into a subnet
    - If a host of the subnet tries to contact the destination host, he will immediately receive an ICMP protocol/port unreachable from the attacker‘s host
    - The host has no possibility to contact the destination host → communication fails
  - Type: DoS

ICMP Destination Unreachable Attack



Attacker

A

ICMP Destination Unreachable response (reason protocol or port) src=172.16.100.12, dest=192.168.1.1

1a-1d

4a-4d

192.168.1.1

3a

ICMP Destination Unreachable response, (reason protocol or port) src=172.16.100.12, dest=192.168.1.1

192.168.1.2

Internet

192.168.1.3

2a-2d

4a-4d

3b

communication between 192.168.1.1 and 172.16.100.12 will fail

B

192.168.1.4

172.16.100.12

Attacks on IP 5

- **Ping sweep**
  - Could be filtered by a packet-level firewall (router)
    - But then you prevent network diagnostic information too
  - Therefore IDS / IPS may be used
    - Ping sweep may be detected by an Intrusion Detection System (IDS) as part of a so called “signature”
    - Ping sweep could be filtered by an Intrusion Prevention System (IPS) in case of emergence
- **Port scan**
  - Active reachable services must have open ports
    - Question: Should they answer or not with ICMP port unreachable message?
  - Again could be filtered by a router to allow only testing of wanted services
  - Again IDS / IPS may be used

What to Do On a Router? 1

- **Disable or restrict all unwanted management services**
  - BootP, TFTP, HTTP, SNMPv1, DNS, NTP, finger
  - CDP (Cisco), Autoloading / Netconfig Booting (Cisco)
  - Echo, chargen, daytime, discard (Cisco’s “service udp/tcp-small-servers”)
- **Disable**
  - IP directed broadcasts
  - IP source route
  - ICMP mask / redirect / unreachable replies on non-trusted interfaces
  - Proxy ARP, Gratuitous ARP (on PPP RAS)

What to Do On a Router? 2

- **RFC 2827 (BCP 38)**
  - Network ingress filtering
    - An ISP allows only packets to enter the Internet with source addresses of own address range
    - Best way to do: outgoing filter at the router interface towards the Internet or incoming filter at all router interfaces towards ISP customers
    - Also included addresses that are reserved for the private IP address range, IP multicast address range, IP experimental address range
      - 0.0.0.0/8, 10.0.0.0/8, 127.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
      - 224.0.0.0/4, 240.0.0.0/4.
    - Against IP spoofing
    - To be a good Internet citizen
- **RFC 3704 (BCP 84)**
  - Ingress filtering for multihomed networks (customers connected to two ISP´s)

Finger (RFC 1288)

- **Is a protocol that returns information on users logged in on a specified hosts**
- **Simple protocol**
  - Finger client connects to TCP port 79 and sends a request
  - Finger server replies with info and closes the connection
    - Login name
    - Name of the user
    - Idle time
    - Office phone number

Finger Attacks

- **Problems:**
  - Finger may provide too much information and therefore may be used for reconnaissance attack
    - List of active users (is user logged in?)
    - Idle time (is user idle?)
    - Source host of the user (possible trust relationship)
  - Forwarding of finger
    - E.g. finger name@ host1@ host2@...
    - A remote server performs the finger command
  - May be used for DoS attacks with finger forwarding
    - Finger username@@@@@@@...@@@host
    - Starts many instances of the finger daemon which possibly exhausts the process table → no new processes can be started
  - Finger forwarding should be disabled

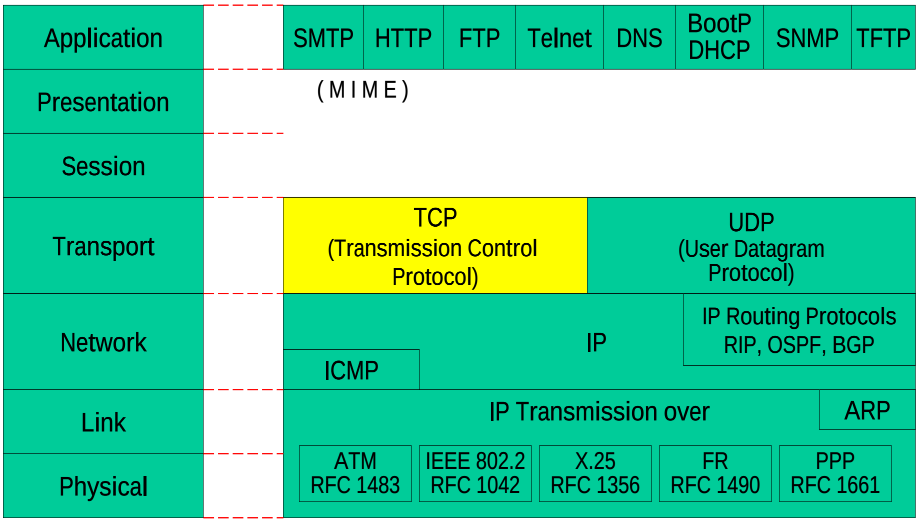
Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

TCP (Transmission Control Protocol)

- **TCP is a connection oriented layer 4 protocol (transport layer) and is transmitted inside the IP data field**
- **Provides a secure end-to-end transport of data between computer processes of different end systems**
- **Secure transport means:**
  - Error detection and recovery
  - Maintaining the order of the data without duplication or loss
  - Flow control
- **RFC 793**

Transport Layer Protocols



Application

Presentation

Session

Transport

Network

Link

Physical

SMTP HTTP FTP Telnet DNS BootP DHCP SNMP TFTP

(MIME)

TCP (Transmission Control Protocol)

UDP (User Datagram Protocol)

IP

IP Routing Protocols RIP, OSPF, BGP

ICMP

IP Transmission over

ARP

ATM RFC 1483

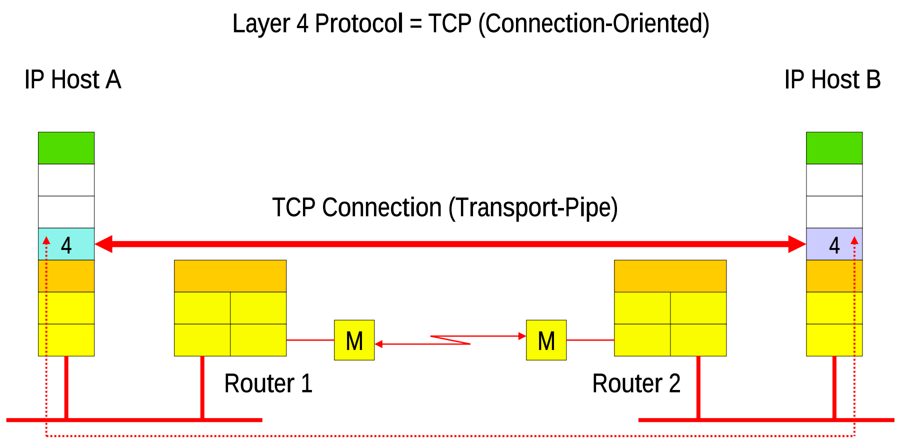
IEEE 802.2 RFC 1042

X.25 RFC 1356

FR RFC 1490

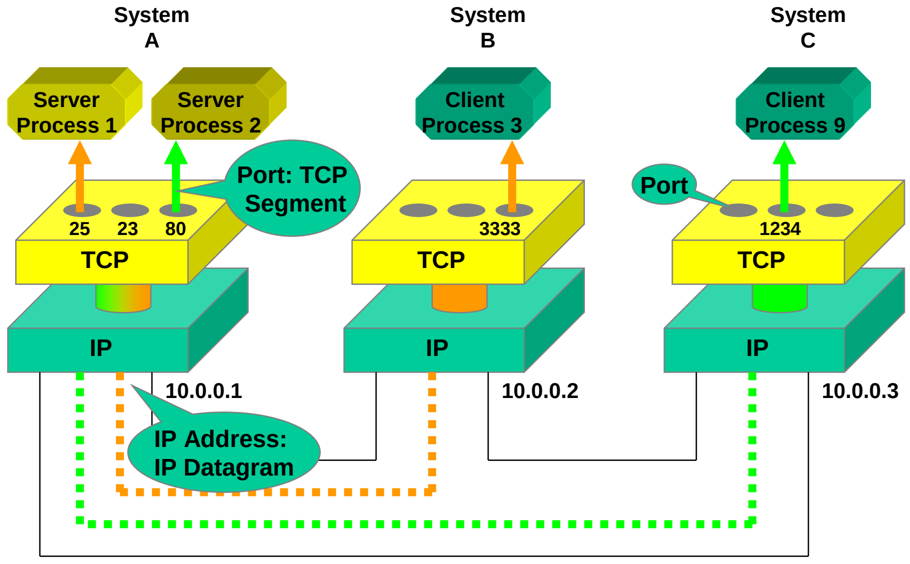
PPP RFC 1661

TCP and OSI Transport Layer 4


Layer 4 Protocol = TCP (Connection-Oriented)
IP Host A
IP Host B
TCP Connection (Transport-Pipe)
4
4
M
M
Router 1
Router 2

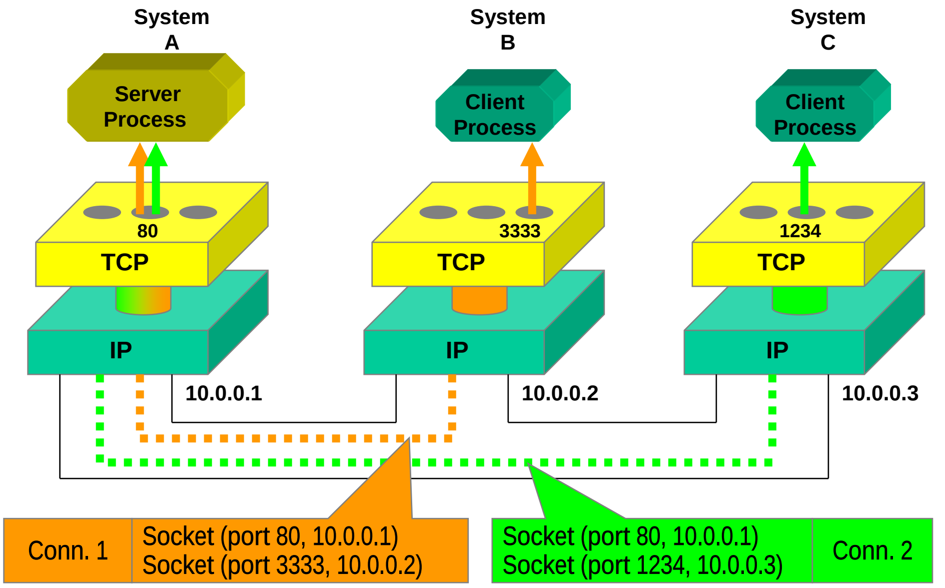
© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 73

TCP Ports and Connections


System A
System B
System C
Server Process 1
Server Process 2
Client Process 3
Client Process 9
Port: TCP Segment
Port
25
23
80
3333
1234
TCP
TCP
TCP
IP
IP
IP
10.0.0.1
10.0.0.2
10.0.0.3
IP Address: IP Datagram

© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 74

TCP Sockets and Connections


System A
Server Process
80
TCP
IP
10.0.0.1
System B
Client Process
3333
TCP
IP
10.0.0.2
System C
Client Process
1234
TCP
IP
10.0.0.3
Conn. 1
Socket (port 80, 10.0.0.1)
Socket (port 3333, 10.0.0.2)
Socket (port 80, 10.0.0.1)
Socket (port 1234, 10.0.0.3)
Conn. 2

TCP Well Known Ports

- **Well known ports**
  - Are reserved for common applications and services (like Telnet, WWW, FTP etc.) and are in the range from 0 to 1023
  - Are controlled by IANA (Internet Assigned Numbers Authority)
  - Server applications listen on their well-known ports for incoming connections
- **Registered ports**
  - start at 1024 (e.g. Lotus Notes, Cisco XOT, Oracle, license managers etc.). They are not controlled by the IANA (only listed, see RFC1700)

TCP User Ports

- **Client applications chose a free port number (which is not already used by another connection) as the source port**
- **The destination port is the well-known port of the server application**
- **Some services like FTP or Remote Procedure Call use dynamically assigned port numbers:**
  - Sun RPC (Remote Procedure Call) uses a portmapper located at port 111...
  - FTP uses the PORT and PASV commands...
- **...to switch to a non-standard port**

Well Known Ports

Some Well Known Ports

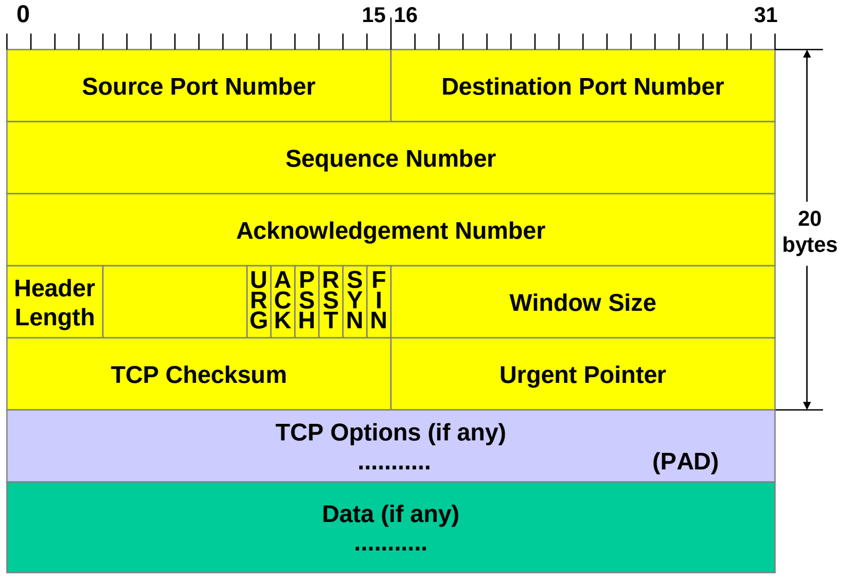
7	**Echo**
20	**FTP (Data), File Transfer Protocol**
21	**FTP (Control)**
23	**TELNET, Terminal Emulation**
25	**SMTP, Simple Mail Transfer Protocol**
53	**DNS, Domain Name Server**
69	**TFTP, Trivial File Transfer Protocol**
80	**HTTP Hypertext Transfer Protocol**
111	**Sun RPC, Sun Remote Procedure Call**
161	**SNMP, Simple Network Management Protocol**
162	**SNMPTRAP**

Some Registered Ports

1416	**Novell LU6.2**
1433	**Microsoft-SQL-Server**
1439	**Eicon X25/SNA Gateway**
1527	**oracle**
1986	**cisco license managmt**
1998	**cisco X.25 service (XOT)**
6000	\
.....	**> X Window System**
6063	**/**

... etc.
(see RFC1700)

TCP Header


0
15
16
31
Source Port Number
Destination Port Number
Sequence Number
Acknowledgement Number
Header Length
URG
ACK
PSH
RST
SYN
FIN
Window Size
TCP Checksum
Urgent Pointer
20 bytes
TCP Options (if any)
....
(PAD)
Data (if any)
....

TCP Header Entries

- **Source and Destination Port**
  - Port number for source and destination process
- **Header Length**
  - Indicates the length of the header given as a multiple of 32 bit words (4 octets)
  - necessary, because of the variable header length

TCP Header Entries

- **Sequence Number (32 Bit)**
  - Position of the first octet of this segment within the data stream ("wraps around" to 0 after reaching 2^32 -1)
- **Acknowledge Number (32 Bit)**
  - Acknowledges the correct reception of all octets up to ack-number minus 1 and indicates the number of the next octet expected by the receiver

TCP Header Entries

- **Flags: SYN, ACK**
  - SYN: If set, the Sequence Number holds the initial value for a new session
    - SYN is used only during the connect phase (can be used to recognize who is the caller during a connection setup e.g. for firewall filtering)
  - Used for call setup (connect request)
  - ACK: If set, the Acknowledge Number is valid and indicates the sequence number of the next octet expected by the receiver

TCP Header Entries

- **Flags: FIN, RST**
  - FIN: If set, the Sequence Number holds the number of the last transmitted octet of this session
    - using this number a receiver can tell that all data have been received; FIN is used only during the disconnect phase
  - Used for call release (disconnect)
  - RST: If set, the session has to be cleared immediately
    - Can be used to refuse a connection-attempt or to "kill" a current connection.

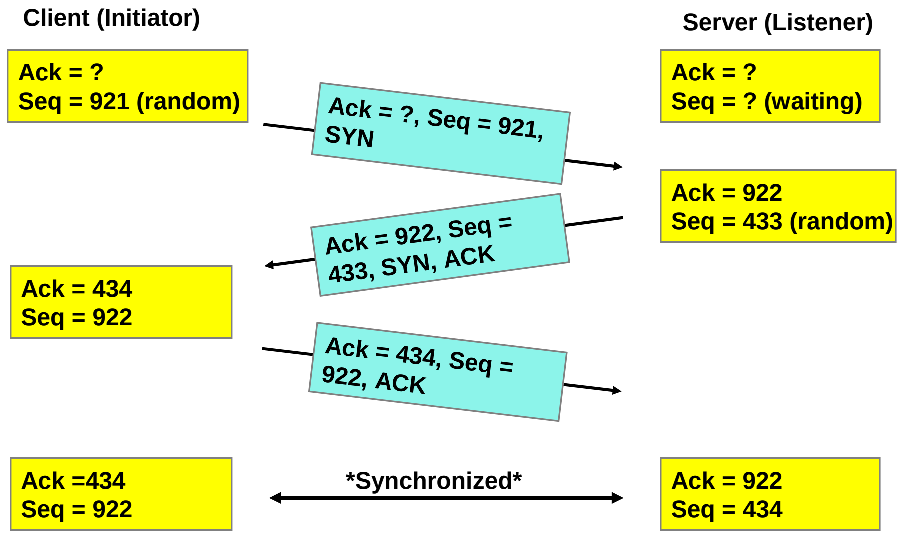
TCP Header Entries

- **Window (16 Bit)**
  - Set by the source with every transmitted segment to signal the current window size; this "dynamic windowing" enables receiver-based flow control
  - The value defines how many additional octets will be accepted, starting from the current acknowledgment number
    - SeqNr of last octet allowed to sent: AckNr plus window value
  - Remarks:
    - Once a given range for sending data was given by a received window value, it is not possible to shrink the window size to such a value which gets in conflict with the already granted range
    - so the window field must be adapted accordingly in order to achieve the flow control mechanism STOP

TCP Header Entries

- **Checksum**
  - The checksum includes the TCP header and data area plus a 12 byte pseudo IP header
    - (one´s complement of the sum of all one´s complements of all 16 bit words)
  - The pseudo IP header contains the source and destination IP address, the IP protocol type and IP segment length (total length). This guarantees, that not only the port but the complete socket is included in the checksum
  - Including the pseudo IP header in the checksum allows the TCP layer to detect errors, which can't be recognized by IP (e.g. IP transmits an error-free TCP segment to the wrong IP end system)

TCP Connect Phase



Client (Initiator)

Server (Listener)

Ack = ?
Seq = 921 (random)

Ack = ?
Seq = ? (waiting)

Ack = ?, Seq = 921, SYN

Ack = 922
Seq = 433 (random)

Ack = 922, Seq = 433, SYN, ACK

Ack = 434
Seq = 922

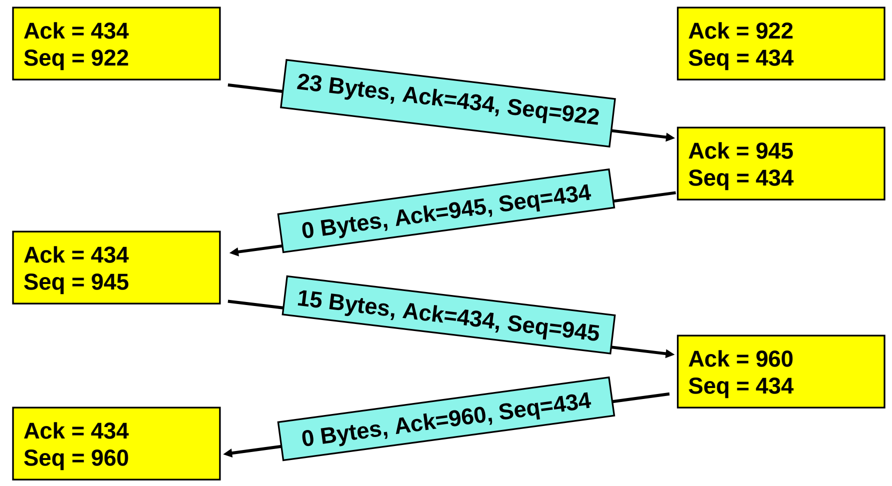
Ack = 434, Seq = 922, ACK

Ack =434
Seq = 922

Synchronized

Ack = 922
Seq = 434

TCP Data Transfer Phase


Ack = 434
Seq = 922
23 Bytes, Ack=434, Seq=922
Ack = 922
Seq = 434
Ack = 945
Seq = 434
0 Bytes, Ack=945, Seq=434
Ack = 434
Seq = 945
15 Bytes, Ack=434, Seq=945
Ack = 960
Seq = 434
0 Bytes, Ack=960, Seq=434
Ack = 434
Seq = 960

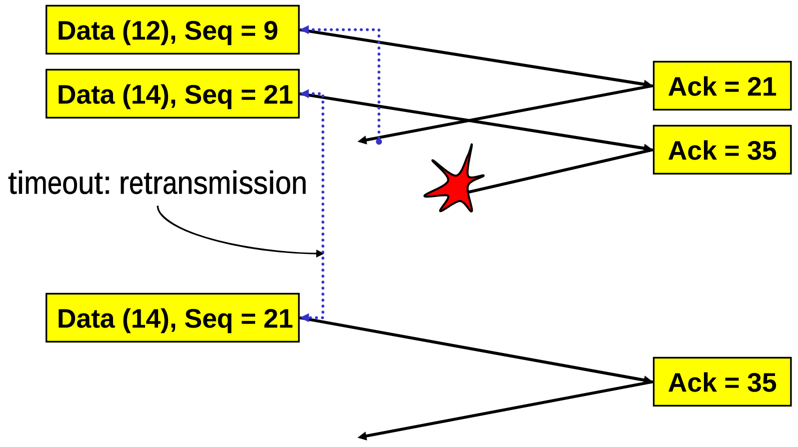
TCP Data Transfer Phase - Error Recovery

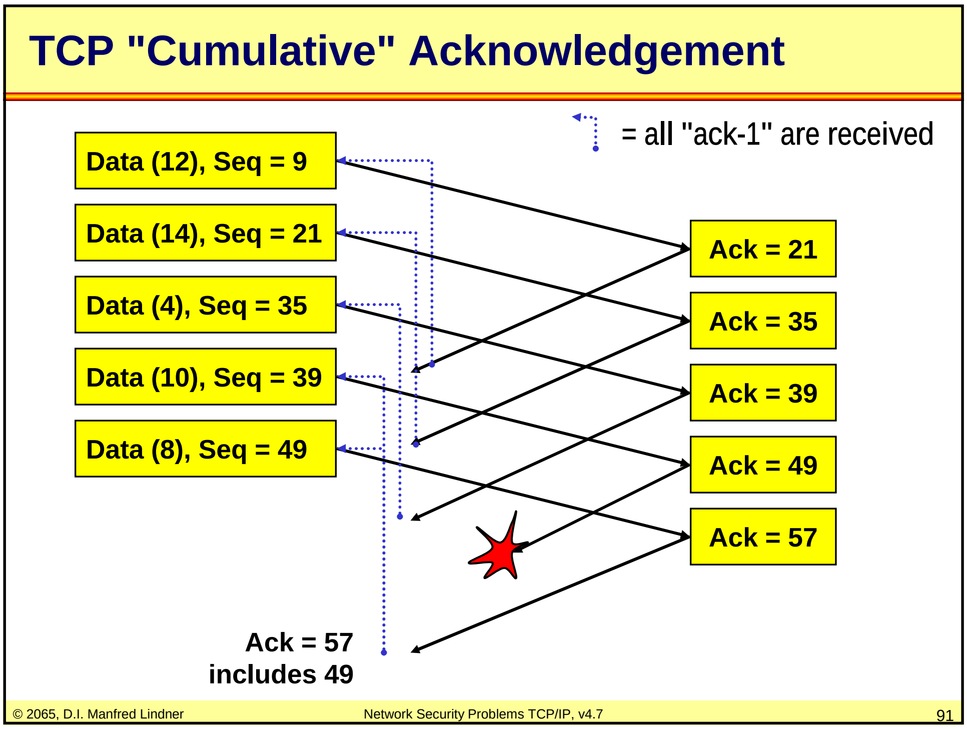
- **Acknowledgements are generated for all octets which arrived in sequence without errors (positive acknowledgement)**
  - Note: duplicates are also acknowledged
  - If a segment arrives out of sequence, no acknowledges are sent until this "gap" is closed
- **The acknowledge number is equal to the sequence number of the next octet to be received**
  - Acknowledges are "cumulative": Ack(N) confirms all octets with sequence numbers up to N-1
  - Thus, lost acknowledgements are not critical since the following ack confirms all previous segments

TCP Timeout

- **Timeout will initiate a retransmission of unacknowledged data**
- **Value of retransmission timeout influences performance (timeout should be in relation to round trip delay)**
  - High timeout results in long idle times if an error occurs
  - Low timeout results in unnecessary retransmissions
- **Adaptive timeout**
  - KARN algorithm uses a backoff method to adapt to the actual round trip delay

TCP Duplicates, Lost Acknowledgement


Data (12), Seq = 9
Data (14), Seq = 21
Ack = 21
Ack = 35
timeout: retransmission
Data (14), Seq = 21
Ack = 35



TCP "Cumulative" Acknowledgement

= all "ack-1" are received

Data (12), Seq = 9

Data (14), Seq = 21

Data (4), Seq = 35

Data (10), Seq = 39

Data (8), Seq = 49

Ack = 21

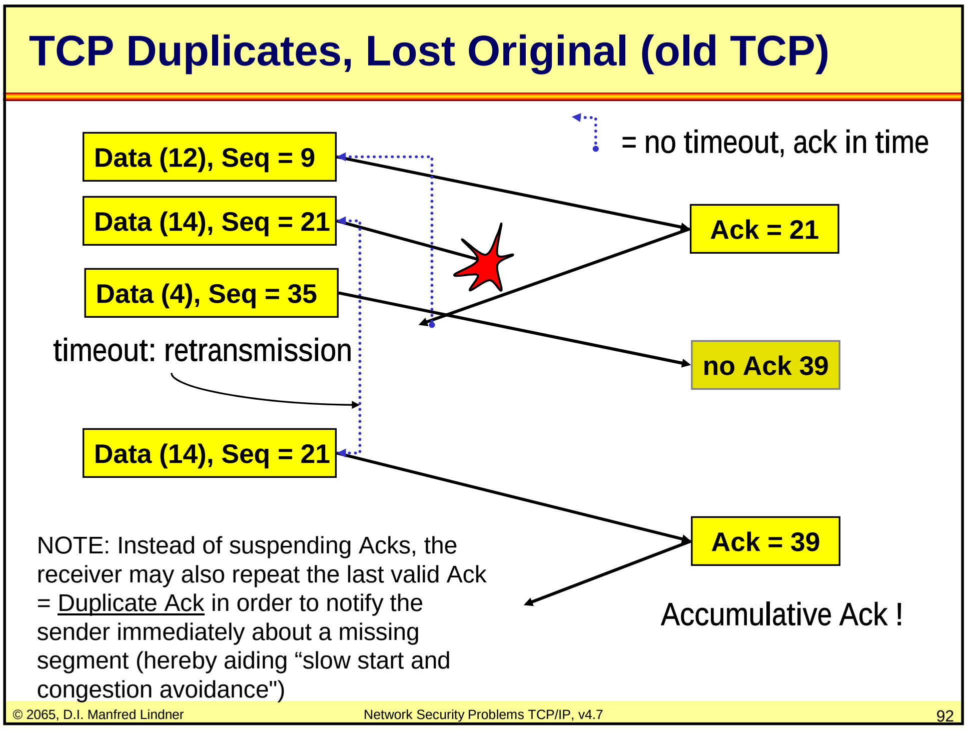
Ack = 35

Ack = 39

Ack = 49

Ack = 57

Ack = 57
includes 49



TCP Duplicates, Lost Original (old TCP)

= no timeout, ack in time

Data (12), Seq = 9

Data (14), Seq = 21

Data (4), Seq = 35

Ack = 21

no Ack 39

timeout: retransmission

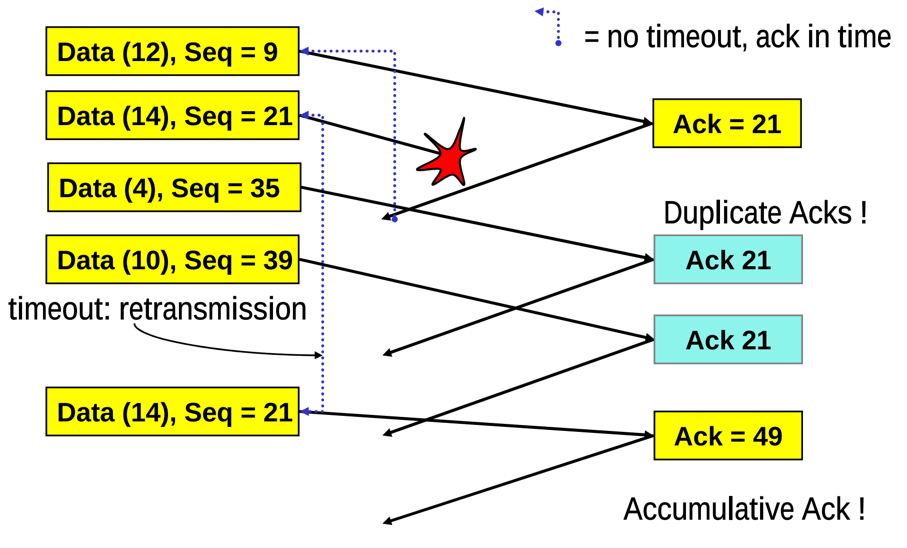
Data (14), Seq = 21

Ack = 39

Accumulative Ack !

NOTE: Instead of suspending Acks, the receiver may also repeat the last valid Ack = Duplicate Ack in order to notify the sender immediately about a missing segment (hereby aiding “slow start and congestion avoidance")

TCP Duplicate-Ack (new TCP)



= no timeout, ack in time

Data (12), Seq = 9

Data (14), Seq = 21

Data (4), Seq = 35

Data (10), Seq = 39

timeout: retransmission

Data (14), Seq = 21

Ack = 21

Duplicate Acks !

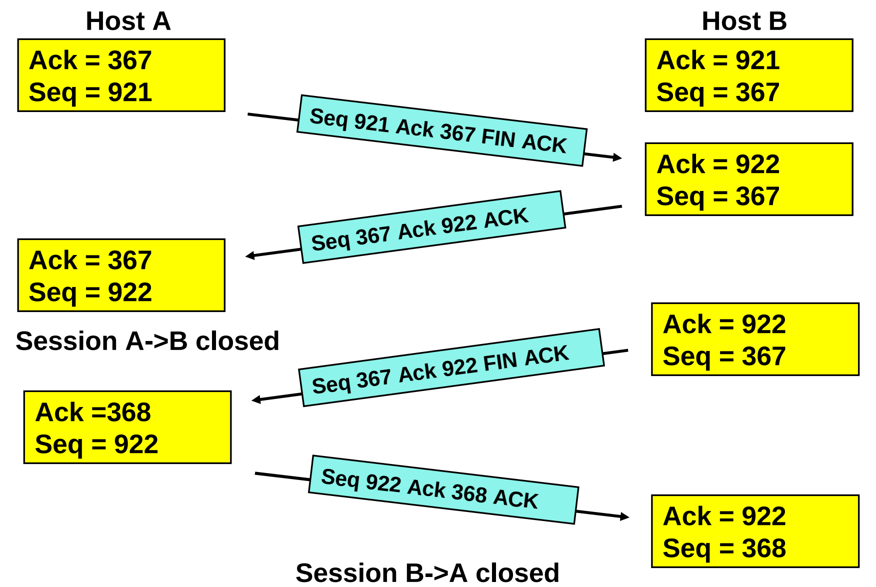
Ack 21

Ack 21

Ack = 49

Accumulative Ack !

TCP Disconnect



Host A

Ack = 367
Seq = 921

Host B

Ack = 921
Seq = 367

Seq 921 Ack 367 FIN ACK

Ack = 922
Seq = 367

Seq 367 Ack 922 ACK

Ack = 367
Seq = 922

Session A->B closed

Ack = 922
Seq = 367

Seq 367 Ack 922 FIN ACK

Ack =368
Seq = 922

Seq 922 Ack 368 ACK

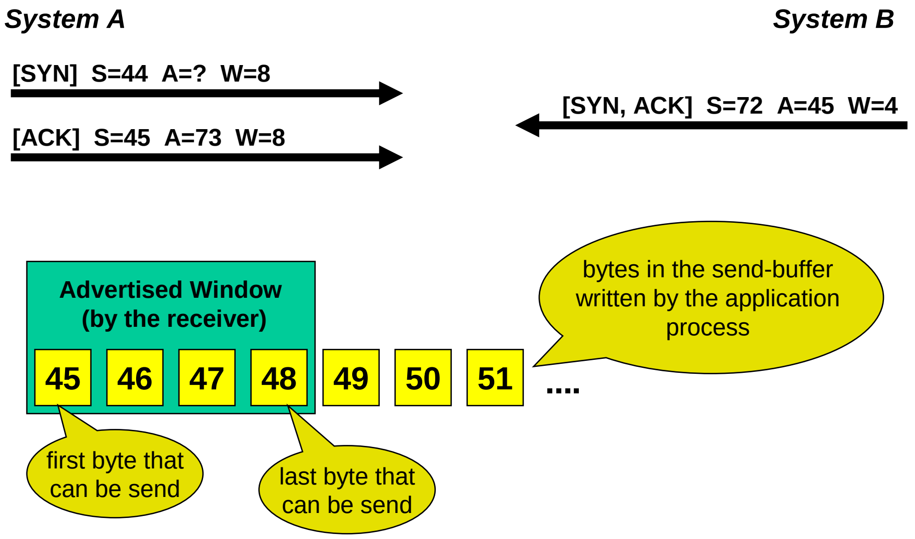
Ack = 922
Seq = 368

Session B->A closed

Flow control: "Sliding Window"

- **TCP flow control is done with dynamic windowing using the sliding window protocol**
- **The receiver advertises the current amount of octets it is able to receive**
  - using the window field of the TCP header
  - values 0 through 65535
- **Sequence number of the last octet a sender may send = received ack-number -1 + window size**
  - The starting size of the window is negotiated during the connect phase
  - The receiving process can influence the advertised window, hereby affecting the TCP performance

Sliding Window: Initialization



System A

System B

[SYN] S=44 A=? W=8

[SYN, ACK] S=72 A=45 W=4

[ACK] S=45 A=73 W=8

Advertised Window (by the receiver)

45 46 47 48 49 50 51 ....

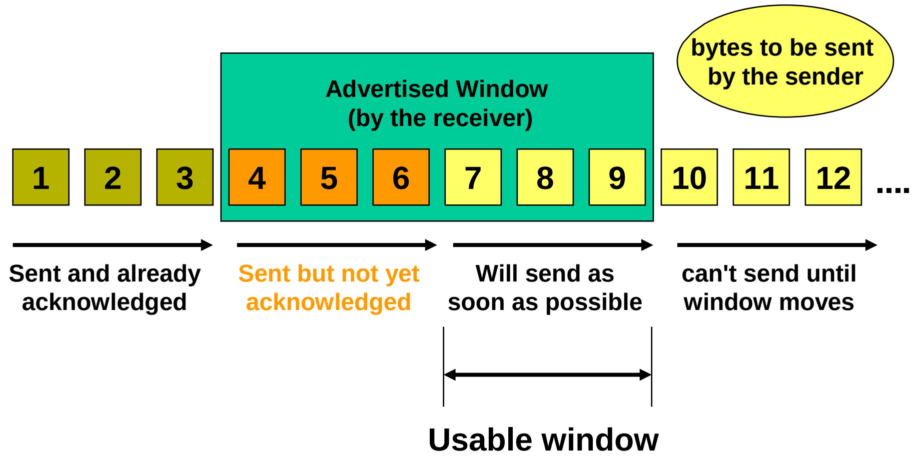
bytes in the send-buffer written by the application process

first byte that can be send

last byte that can be send

Sliding Window: Principle

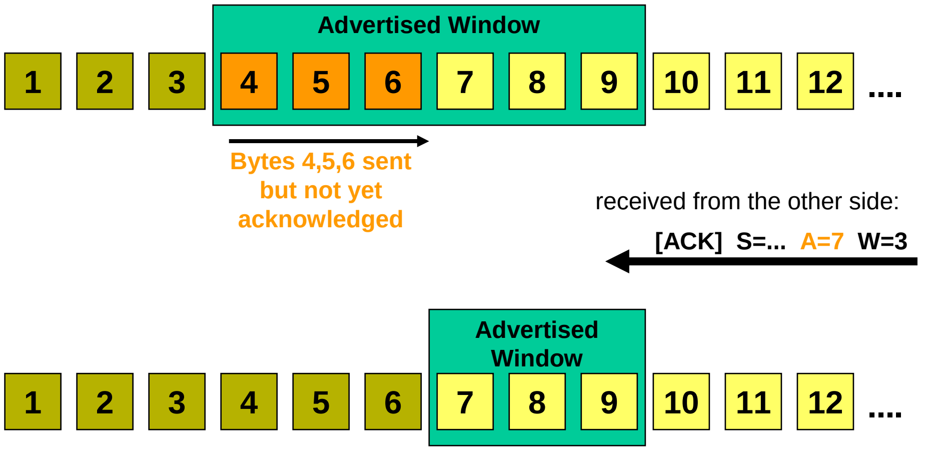
Sender's point of view; sender got {ACK=4, WIN=6} from the receiver.


bytes to be sent by the sender
Advertised Window (by the receiver)
1
2
3
4
5
6
7
8
9
10
11
12
....
Sent and already acknowledged
Sent but not yet acknowledged
Will send as soon as possible
can't send until window moves
Usable window

Sliding Window

- **During the transmission the sliding window moves from left to right, as the receiver acknowledges data**
- **The relative motion of the two ends of the window *open* or *closes* the window**
  - the window closes when data - already sent - is acknowledged (the left edge advances to the right)
  - the window opens when the receiving process on the other end reads data - and hence frees up TCP buffer space - and finally acknowledges data with a appropriate window value (the right edge moves to the right)
- **If the left edge reaches the right edge, the sender stops transmitting data - *zero usable window***

Closing the Sliding Window


Advertised Window
1
2
3
4
5
6
7
8
9
10
11
12
....
Advertised Window
1
2
3
4
5
6
7
8
9
10
11
12
....

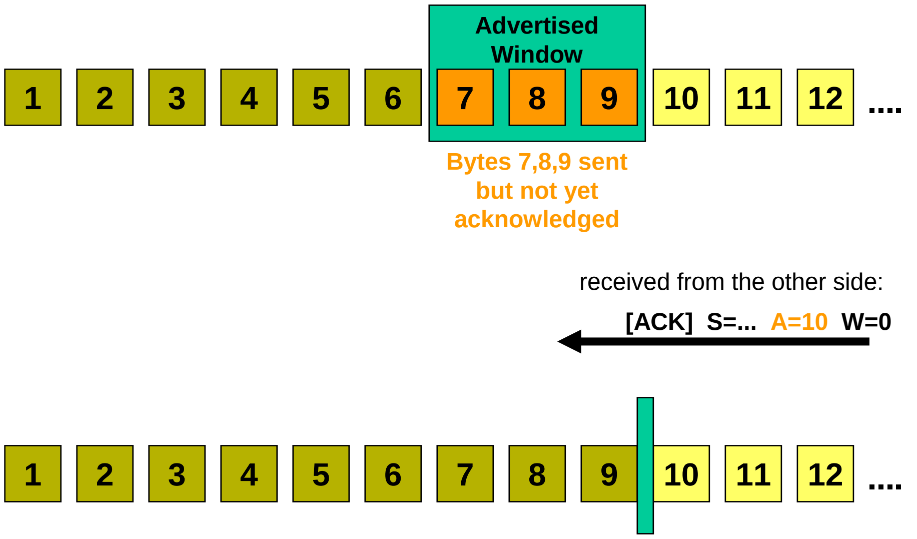
Bytes 4,5,6 sent but not yet acknowledged

received from the other side:

[ACK] S=... A=7 W=3

Now the sender may send bytes 7, 8, 9. The receiver didn't open the window (W=3, right edge remains constant) because of congestion. However, the remaining three bytes inside the window are already granted, so the receiver cannot move the right edge leftwards.

Flow Control -> STOP, Window Closed

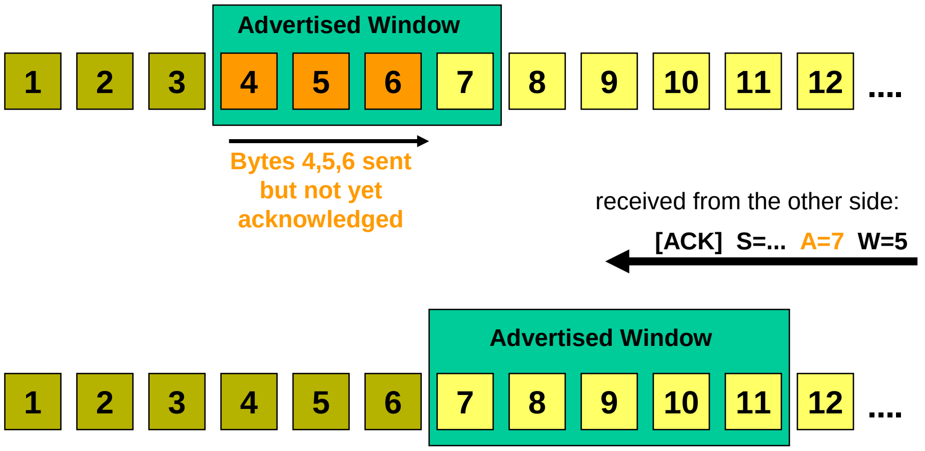

Advertised Window
1
2
3
4
5
6
7
8
9
10
11
12
....
1
2
3
4
5
6
7
8
9
10
11
12
....

Bytes 7,8,9 sent but not yet acknowledged

received from the other side:

[ACK] S=... A=10 W=0

Opening the Sliding Window


Advertised Window
1
2
3
4
5
6
7
8
9
10
11
12
....
Bytes 4,5,6 sent
but not yet
acknowledged
received from the other side:
[ACK] S=... A=7 W=5
Advertised Window
1
2
3
4
5
6
7
8
9
10
11
12
....

The receiver's application read data from the receive-buffer and acknowledged bytes 4,5,6. Free space of the receiver's buffer is indicated by a window value that makes the right edge of the window move rightwards. Now the sender may send bytes 7, 8, 9,10,11.

TCP Enhancements

- **So far, only basic TCP procedures have been mentioned**
- **TCP's development still continues; it has been already enhanced with additional functions which are essential for operation of TCP sessions in today's IP networks:**
  - Slow Start and Congestion Avoidance Mechanism
  - Fast Retransmit and Fast Recovery Mechanism
  - Delayed Acknowledgements
  - The Nagle Algorithm
  - ....

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

Main Network Security Problems with TCP

- **TCP**
  - TCP (IP) Spoofing / TCP Hijacking
    - Sequence number attack
    - Authentication based on IP source address
  - DoS (Denial of Service)
    - Open many TCP connections to one machine at the same time with forged IP source address
      - SYN flooding
    - Mail bombing
  - DDos (Distributed Denial of Service)
    - DoS attack started from different often innocent machines (zombies or drones) at the same time; zombie code installed by virus

Famous TCP Attacks

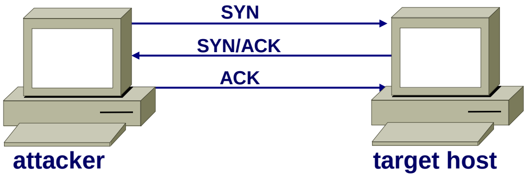
- **TCP Fragmentation**
- **TCP Scanning**
- **OS Fingerprinting**
- **TCP Spoofing**
- **TCP Hijacking**
- **TCP SYN Flooding**

Fragmentation – Attack

- **TCP overwrite (fragmentation overlap)**
  - IP datagram containing TCP traffic is fragmented
  - First fragment contains TCP header with allowed port (e.g. 25) => stateless firewalls will let this packet pass
  - But next fragments contain no TCP header
  - The trick is to set the value of the fragment offset on the second packet so low, that instead of appending the second packet to the first packet, it actually overwrites the data and part of the TCP header of the first packet
  - After packet has been reassembled completely at the end-system, it will be delivered to a new port (which normally may not pass the firewall if sent without fragments)
- **Alternative (tiny fragment)**
  - Fragment between IP and TCP header (before TCP port fields)

TCP Scanning

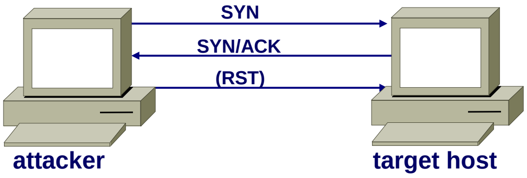
- **Used to check whether a port on a host is open**
- **Used to get some extra information about the host**
- **How to test a open port?**



- **“Vanilla” TCP scan**
- **But from the hackers view**
  - Should be done without informing the host that it is scanned

TCP SYN Scanning

- **Also known as „Half-open“ or SYN scan**



  - The attacker sends a SYN packet
  - If host answers with SYN/ACK the port is open
  - The attacker sends either nothing or a RST packet instead of an ACK
- **Advantage for the hacker:**
  - The connection is never opened and the event may not be logged by the operating system / monitor application

TCP FIN Scanning

- **The attacker sends a FIN-marked packet**
  - In most TCP/IP implementations (not Windows)
    - If the port is closed a RST packet is sent back
    - If the port is open the FIN packet is ignored
- **A lot of other types of this scanning technique exists:**
  - XMAS scan: FIN + PSH + URG flag set
  - NULL scan: no flags set
  - ACK scan: to avoid three way handshake start sequence
- **Useful tool**
  - Nmap
    - http://www.insecure.org/nmap
  - Also may be used for OS fingerprinting

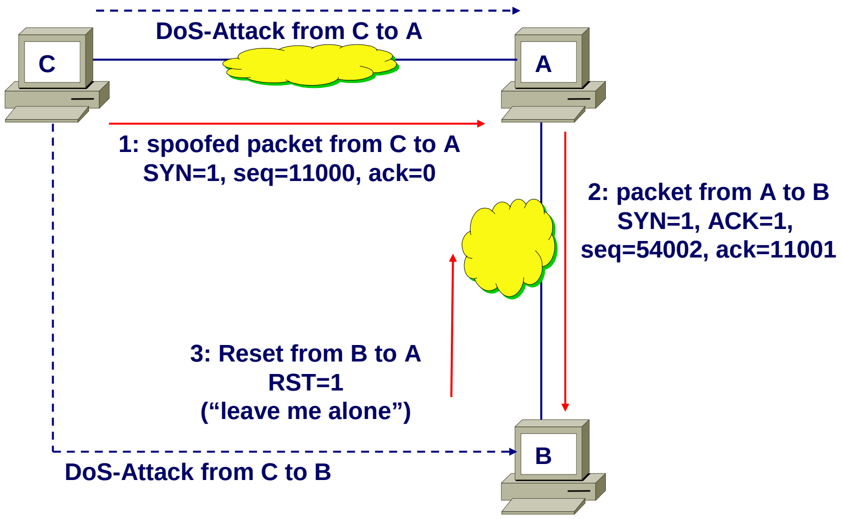
TCP Fingerprinting / OS Fingerprinting

- **Every OS has its own TCP/IP implementation**
- **TCP fingerprinting allows**
  - To determine the operating system of a host by examining the reaction to carefully crafted packets
- **TCP / OS fingerprinting checks the**
  - The behavior to the already mentioned TCP scan techniques
  - Use of reserved flags in the TCP header
  - Selection of TCP initial sequence numbers
  - Response to particular ICMP messages
  - Server response at a special port

TCP (IP) Spoofing Basic Attack 1

- **TCP connection setup by intruder**
  - With a spoofed IP address as source address and a legitimate IP address as the destination address
- **TCP acknowledgement by legitimate station**
  - Answer will be seen by the spoofed station
    - But this machine have never initiated that connection
    - Will reset the connection
- **DoS attack only**
  - Targeted to two machines
  - Tracks of intruder are covered because of IP spoofing
- **For real intrusion not very useful**
  - You cannot have true interactive sessions with a host using this technique because answers will not seen by the intruder
    - Destination based routing
    - Packet will not find the way back to the intruder
  - Exception:
    - Intruder can sniff the packets on a LAN

TCP Spoofing Basic DoS Attack 2


DoS-Attack from C to A

1: spoofed packet from C to A
SYN=1, seq=11000, ack=0

2: packet from A to B
SYN=1, ACK=1,
seq=54002, ack=11001

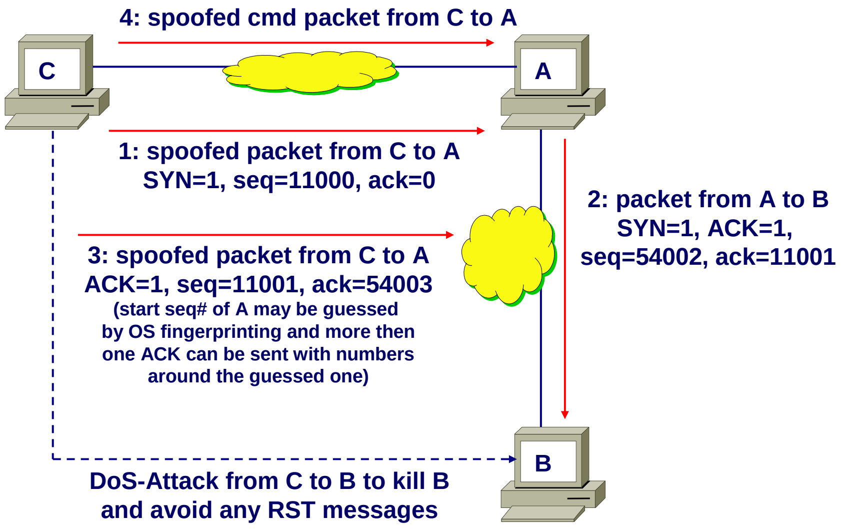
3: Reset from B to A
RST=1
("leave me alone")

DoS-Attack from C to B

TCP Spoofing Advanced Attack 1

- **Prerequisite for this kind of attack are**
  - A trust relationship based on IP address
    - host A trusts host B (e.g. B has successfully logged in and B´s IP address is trusted)
  - Random TCP sequence number may be predicted by probing host A
- **During the TCP connection establishment phase**
  - host C (intruder) kills host B
    - e.g. DoS-attack like SYN flooding, redirecting
  - C sends A a TCP segment with IP spoofing (the source address of B) and an initial sequence number X
  - A replies to B with SYN/ACK and Y as A´s sequence number as well as X+1 as the acknowledge number
  - C does not receive this segment, but it has to send an ACK segment (Y+1) to finish the handshake and setup a one-way connection
  - C can now feed “blindly” host A with some “useful” commands (A thinks it is trusted B); of course answers will still get to B

TCP Spoofing Advanced Attack 2



4: spoofed cmd packet from C to A

1: spoofed packet from C to A
SYN=1, seq=11000, ack=0

2: packet from A to B
SYN=1, ACK=1,
seq=54002, ack=11001

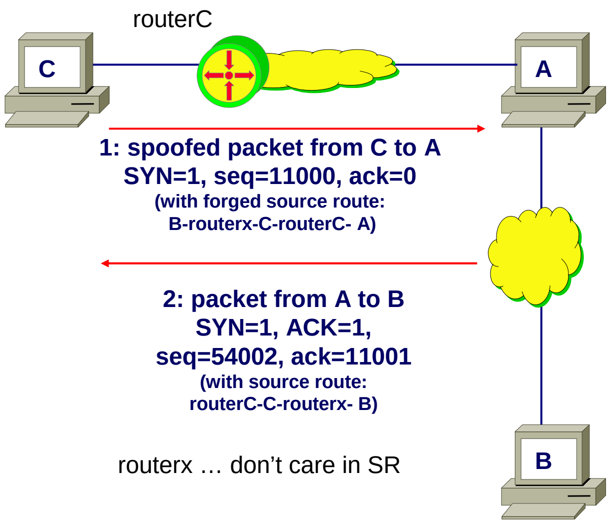
3: spoofed packet from C to A
ACK=1, seq=11001, ack=54003
(start seq# of A may be guessed by OS fingerprinting and more then one ACK can be sent with numbers around the guessed one)

DoS-Attack from C to B to kill B and avoid any RST messages

TCP (IP) Spoofing Attack with SR 1

- **Perform advanced attack with IP spoofing and trusted relationship but use**
  - IP Source Routing Option (SR) instead
- **Then you successfully impersonate the trusted machine**
- **Therefore**
  - Disable this kind of IP source routing in a routed network
  - Avoid trust relationship based on IP address
    - At least outside a firewall protected area
  - Make ISN really random numbers
  - Replace weak “r” commands with ssh, scp, etc.
  - Antispoofing filters at border routers/switches

TCP (IP) Spoofing Attack with SR 2



1: spoofed packet from C to A
SYN=1, seq=11000, ack=0
(with forged source route:
B-routerx-C-routerC- A)

2: packet from A to B
SYN=1, ACK=1,
seq=54002, ack=11001
(with source route:
routerC-C-routerx- B)

routerx … don’t care in SR

no DoS of B is necessary !!!

TCP Hijacking 1

- **Technique to take control of an existing TCP connection**
- **Prerequisite is**
  - The traffic between the client and server must pass by at the intruders machine
  - This may not be the LAN where the client or server is located
    - In such a case the already explained ARP Spoofing with appropriate SW at the intruders machine is sufficient to hijack traffic (= takeover of traffic)
- **Focus on**
  - telnet, ftp, r-
- **How it works:**
  - Spoofed TCP segments are used in order to
    - Insert data into streams ("null data desynchronization")
      or
    - Reset a TCP connection and reopen them in the early stage -> after SYN, SYN/ACK ("early desynchronization")
    - Correct sequence/acknowledge numbers must be used

TCP Hijacking 2

- **The attacker waits until the connection is quiet and data has been acknowledged**
- **The TCP session is synchronized at that stage**
- **Synchronization means**
    - client_SEQ# = server_ACK#
    - server_SEQ# = client_ACK#
- **If now a TCP data segment is received at the server side**
  - the sequence number of this segment (the seen client_SEQ# in the segment) must fit into the range to be accepted
    - [server_ACK#, server_ACK# + server_Window]

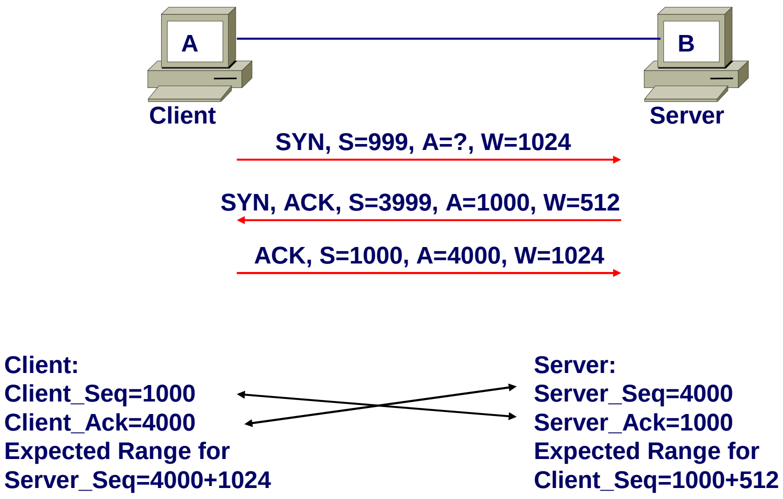
TCP Hijacking 3

- **If now a TCP data segment is received at the client side**
  - – the sequence number of this segment (the seen server_SEQ#) must fit into the range to be accepted
    - [client_ACK#, client_ACK# + client_Window]

- **But if a TCP data segment is received**
  - – and the sequence number does not fit into the range
- **Or if a TCP ACK segment is received**
  - – acknowledging data never sent
- **Then a TCP ACK segment will be sent**
  - – to point to the really expected next sequence number
- **Note: that means desynchronization in both cases**

TCP Hijacking 4

Synchronization after TCP set-up or in an idle phase with all flown data already be acknowledged



SYN, S=999, A=?, W=1024

SYN, ACK, S=3999, A=1000, W=512

ACK, S=1000, A=4000, W=1024

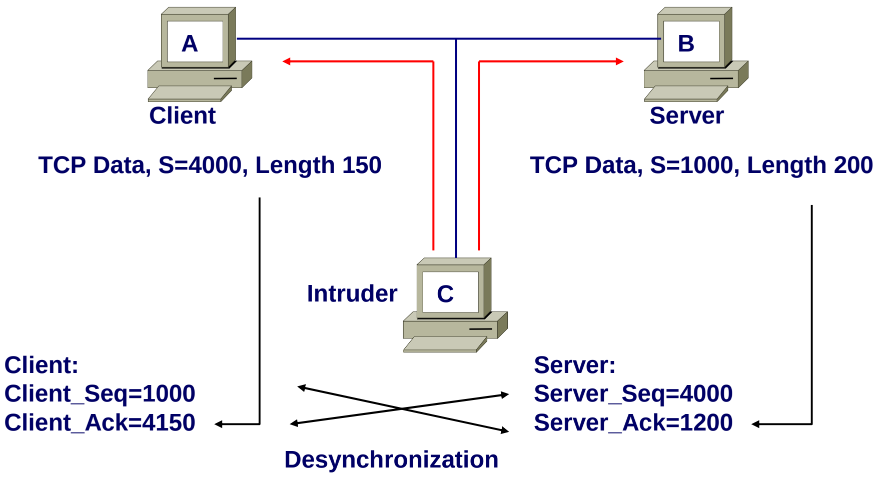
Client:
Client_Seq=1000
Client_Ack=4000
Expected Range for
Server_Seq=4000+1024

Server:
Server_Seq=4000
Server_Ack=1000
Expected Range for
Client_Seq=1000+512

TCP Hijacking 5

- **Now a sufficient amount of null data is injected by the intruder into the TCP stream ("null data desynchronization")**
  - To disturb the synchronization of the TCP connection
  - Null data means that this will not be recognized by the user of the TCP session
    - E.g. in Telnet use IAC NOP, IAC NOP, …. (Telnet command for no operation)
- **This is done by the intruder towards both ends**
  - Server and client
  - Both get desynchronized

TCP Hijacking 6



A

Client

B

Server

TCP Data, S=4000, Length 150

TCP Data, S=1000, Length 200

Intruder C

Client:
Client_Seq=1000
Client_Ack=4150

Server:
Server_Seq=4000
Server_Ack=1200

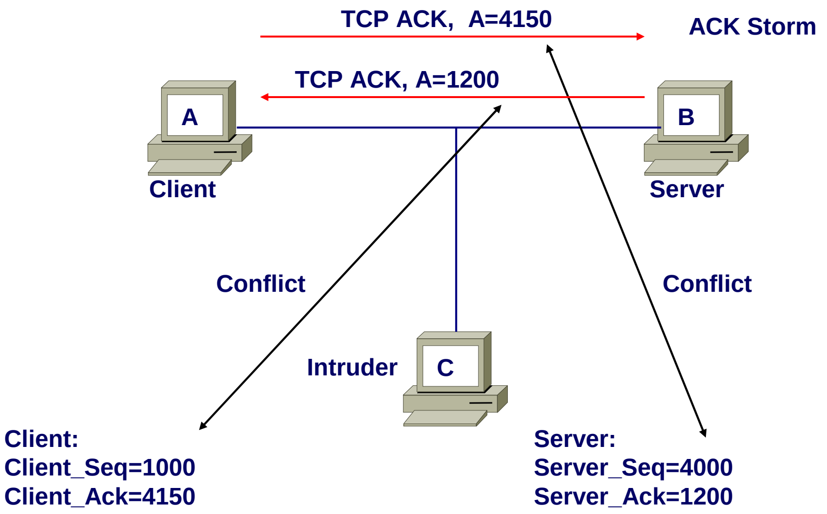
Desynchronization

TCP Hijacking 7

- **Because of desynchronization**
  - client and server send a TCP ACK to the other side
- **Client <-> Server**
  - get very confused and try to resolve the unsynchronized sequence numbers
- **As a result**
  - a so called ACK storm will rise

- **The same result can be achieved**
  - if intruder just insert a spoofed TCP data segments to the server and then the client starts to send a data segment to the server
  - the client data segment is dropped by the server because the sequence number was already seen (from server's the point of view that must be a retransmission of an already received data segment) and an ACK segment is sent from the server to the client to point to the expected value
  - But the ACK number received from the server does not fit (because acknowledges something never sent), so the wrong ACK from the server generates an another ACK packet at the client side

TCP Hijacking 8



TCP ACK, A=4150

ACK Storm

TCP ACK, A=1200

Client:
Client_Seq=1000
Client_Ack=4150

Server:
Server_Seq=4000
Server_Ack=1200

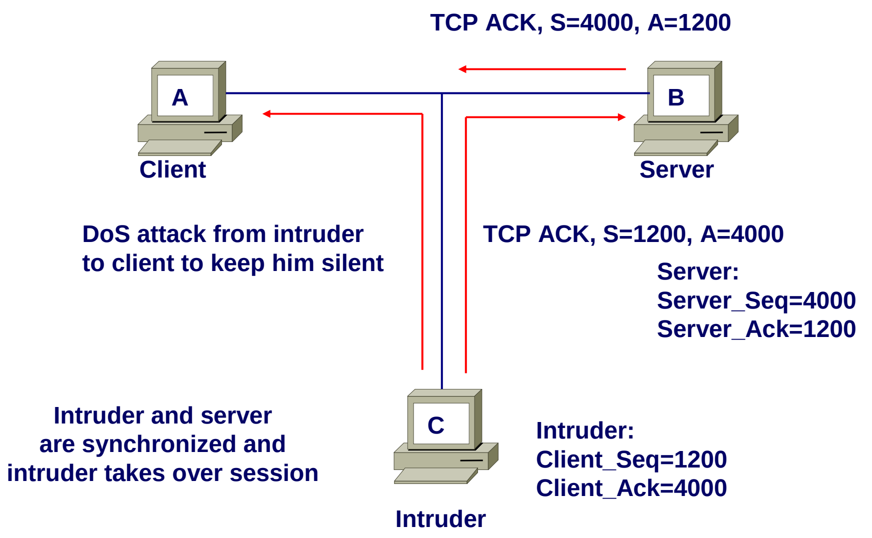
TCP Hijacking

9

- **ACK storm creates an infinite loop**
  - would disturb the takeover of the session by the intruder and also would cause huge network traffic
- **Therefore ACK storm should be avoided**
  - either the intruder performs a DoS attack against the client to keep him silent and synchronize with the server
  - or the intruder is able to become a “man-in-the-middle” (e.g. by ARP spoofing) and such ACK messages are never seen by the real session members
- **As final result the intruder takes over the current client session**
  - client believes that session has died
  - server just sees an ongoing session
    - even one-time passwords used for authentication at TCP session start would fail
    - therefore authentication and integrity check for every single segment is necessary to avoid that attack

TCP Hijacking

10



TCP ACK, S=4000, A=1200

DoS attack from intruder to client to keep him silent

TCP ACK, S=1200, A=4000

Server:
Server_Seq=4000
Server_Ack=1200

Intruder and server are synchronized and intruder takes over session

Intruder:
Client_Seq=1200
Client_Ack=4000

TCP Hijacking 11

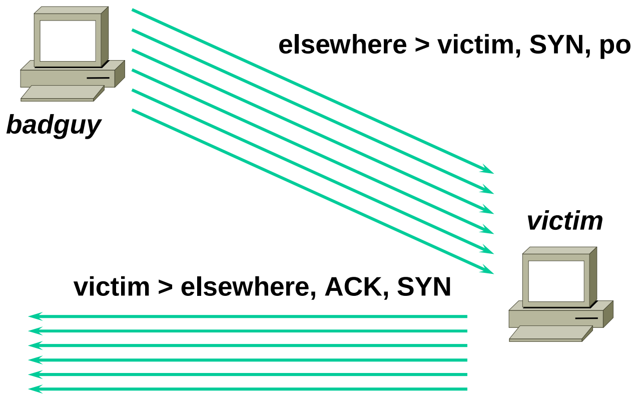
- **Tool for Hijacking:**
  - Hunt
    - written by Pavel Krauz
    - http://www.l0t3k.org/security/tools/hijacking
  - Can perform described scenario
  - Additionally can do ARP spoofing
    - In order to avoid desynchronization, ACK storm and DoS by directing the traffic to the intruders machine and intercepting it
    - Perfect in a LAN environment
    - Also possible if ARP spoofing against routers is performed at the LAN where the intruder resides
    - Only problem for the intruder in such a case
      - Must handle all the router traffic
      - Performance of the intrude machine may be to low
      - Intruder may be recognized
- **Real solution against hijacking**
  - End-system to end-system security (IPsec or SSL/TLS)

TCP DoS Attacks 1

- **TCP SYN Flooding**
  - Very common denial-of-service attack
  - Often combined with IP spoofing
  - Attacker starts handshake with SYN marked segment
  - Victim replies with SYN-ACK segment
  - Attacker‘s host stays silent
    - A host can only keep a limited number of TCP connections in half-open state
    - After that limit, connections are not accepted
  - Current solution:
    - Drop half open connections

TCP SYN Flooding with IP Spoofing

- Locks up the TCP layer software, with only a few packets
- TCP intercept can be used to monitor SYN requests in the Backlog Queue


elsewhere > victim, SYN, port 80
badguy
victim
victim > elsewhere, ACK, SYN

Backlog Queue	
#1	SYN-RECEIVED
#2	SYN-RECEIVED
#3	SYN-RECEIVED
#4	SYN-RECEIVED
#5	SYN-RECEIVED
#6	SYN-RECEIVED

TCP DoS Attacks 2

- **Process Table Attack**
  - Daemons are programs that listen on a particular port for connection requests
  - When a new connection is established the daemon
    - forks a new process that will handle the connection
    - waits for the next connection
  - Many daemons run with root privileges (no restrictions)
    - A huge number of connections fill up the process table -> no new processes can be created

TCP DoS Attacks 3

- **Land Attack**
  - TCP segment with the SYN flag set is sent to an open port
    - But the source address/port is the same as the destination address/port
  - The host tries to open
    - Connection to himself to a port that is already in use
    - Sequence numbers (a new one is generated when the SYN segment arrives) don‘t match which results desynchronization
  - The result is
    - An internal ACK storm, which is very CPU intensive and let OS´s crash

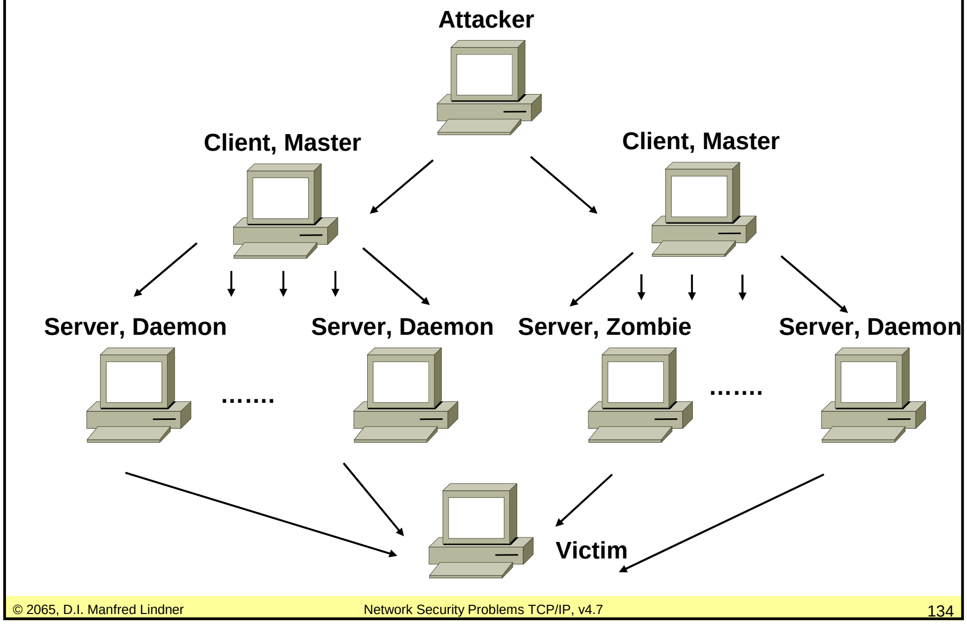
Distributed DoS Attacks 1

- **Perform DoS from more than one evil machine**
- **Especially dangerous**
  - If part of Trojan programs which are synchronized for performing the actual attack
- **Lot of tools:**
  - Tribe Flood Network (TFN) and Tribe Flood Network 2000 (TFN2K)
  - mstream
  - Shaft
  - Stacheldraht
  - Trin00, and the related WinTrin00, Freak88

Distributed DoS Attacks 2

- **Some terminology and techniques**
  - Attacking machines are called daemons / slaves / zombies / agents / servers
    - are usually poorly secured machines that are compromised
  - Machines that control and command the zombies are called masters / handlers / clients
  - Attacker hides himself behind machines that are called stepping stones which control the clients
  - ICMP echo reply messages are used for communication among the beasts (TFN2K)
    - Such traffic was considered as harmless before TNF2K
  - IP Spoofing used by all of them zombies, clients, stepping stones
  - Hard to trace such an event

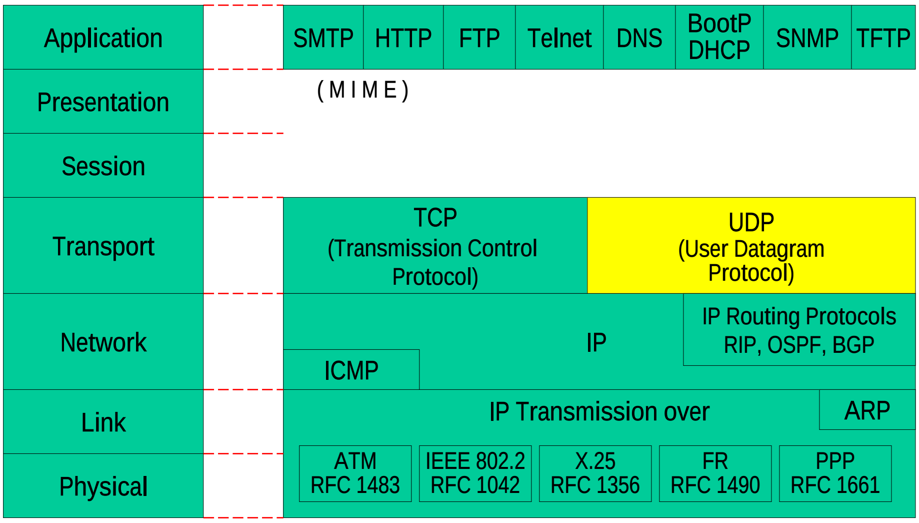
Distributed DoS Attacks 3


Attacker
Client, Master
Client, Master
Server, Daemon
Server, Daemon
Server, Zombie
Server, Daemon
....
....
Victim

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

Transport Layer Protocols



Application

Presentation

Session

Transport

Network

Link

Physical

SMTP HTTP FTP Telnet DNS BootP DHCP SNMP TFTP

(M I M E)

TCP (Transmission Control Protocol)

UDP (User Datagram Protocol)

IP

IP Routing Protocols RIP, OSPF, BGP

ICMP

IP Transmission over

ARP

ATM RFC 1483

IEEE 802.2 RFC 1042

X.25 RFC 1356

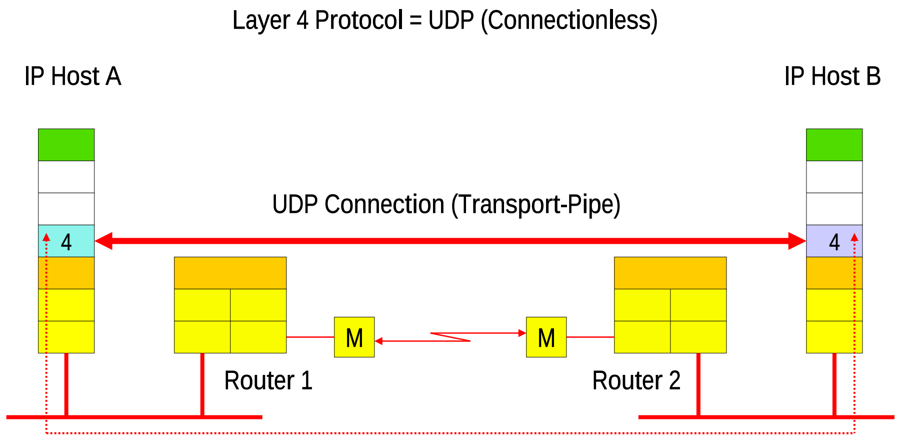
FR RFC 1490

PPP RFC 1661

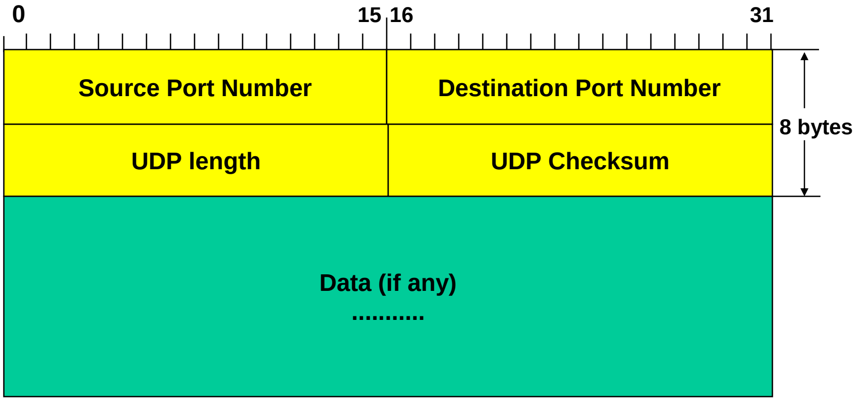
UDP (User Datagram Protocol, RFC 768)

- **UDP is a connectionless layer 4 service (datagram service)**
- **Layer 3 Functions are extended by port addressing and a checksum to ensure integrity**
- **UDP uses the same port numbers as TCP (if applicable)**
- **Less complex than TCP, easier to implement**

UDP and OSI Transport Layer 4


Layer 4 Protocol = UDP (Connectionless)
IP Host A
IP Host B
UDP Connection (Transport-Pipe)
4
4
M
M
Router 1
Router 2

UDP Header



Source Port Number

Destination Port Number

UDP length

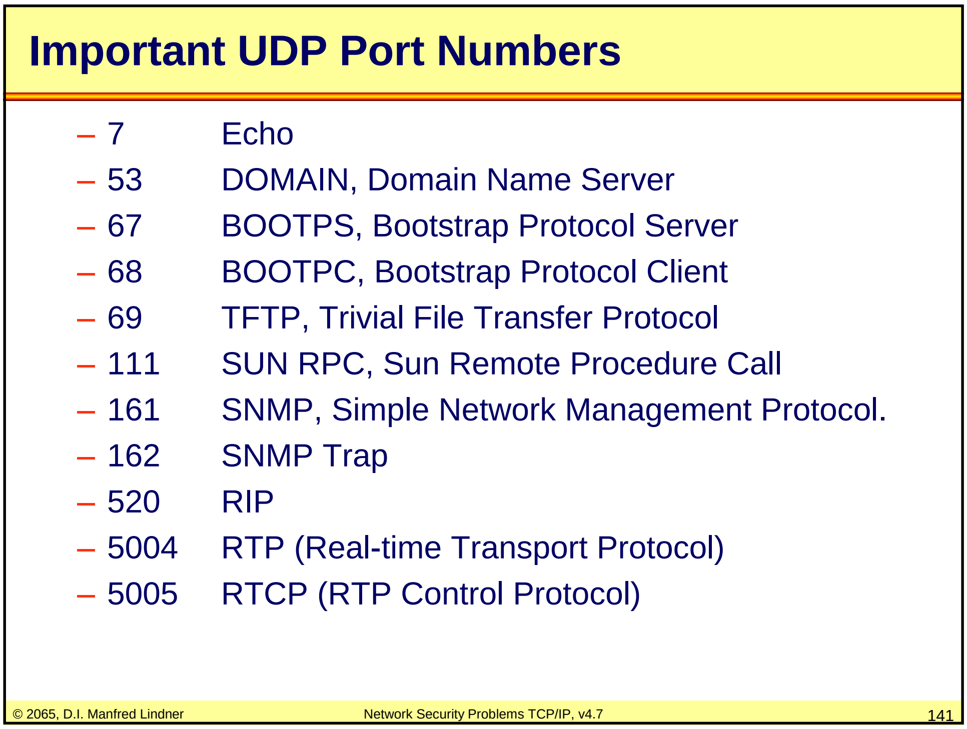
UDP Checksum

Data (if any)

....

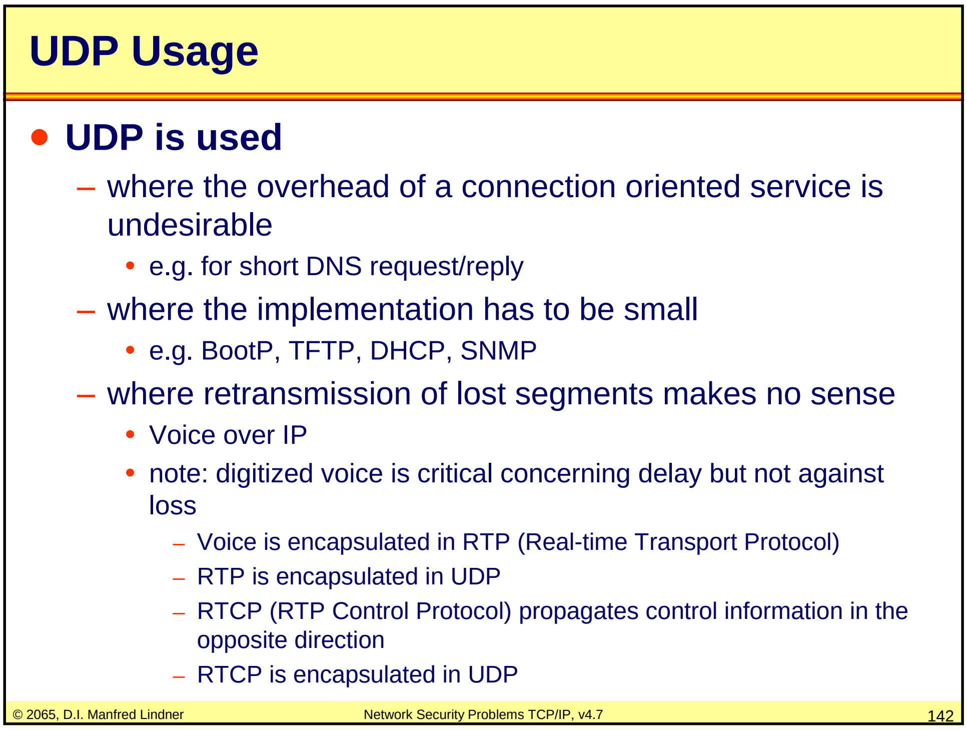
UDP Header Entries

- **Source and Destination Port**
  - Port number for addressing the process (application)
  - Well known port numbers defined in RFC1700
- **UDP Length**
  - Length of the UDP datagram (Header plus Data)
- **UDP Checksum**
  - Checksum includes pseudo IP header (IP src/dst addr., protocol field), UDP header and user data. One´s complement of the sum of all one´s complements



Important UDP Port Numbers

7	Echo
53	DOMAIN, Domain Name Server
67	BOOTPS, Bootstrap Protocol Server
68	BOOTPC, Bootstrap Protocol Client
69	TFTP, Trivial File Transfer Protocol
111	SUN RPC, Sun Remote Procedure Call
161	SNMP, Simple Network Management Protocol.
162	SNMP Trap
520	RIP
5004	RTP (Real-time Transport Protocol)
5005	RTCP (RTP Control Protocol)



UDP Usage

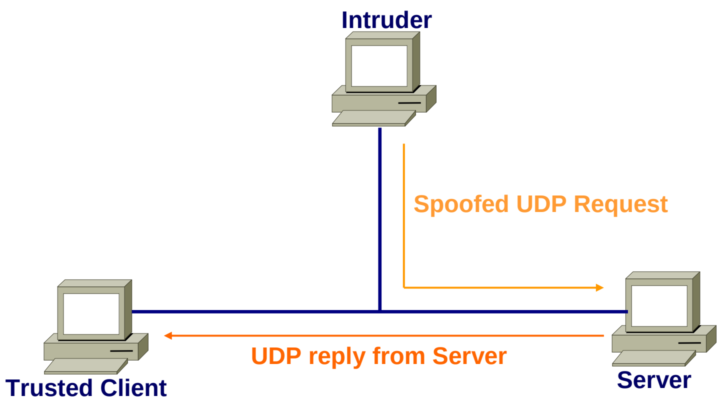
- **UDP is used**
  - where the overhead of a connection oriented service is undesirable
    - e.g. for short DNS request/reply
  - where the implementation has to be small
    - e.g. BootP, TFTP, DHCP, SNMP
  - where retransmission of lost segments makes no sense
    - Voice over IP
    - note: digitized voice is critical concerning delay but not against loss
      - Voice is encapsulated in RTP (Real-time Transport Protocol)
      - RTP is encapsulated in UDP
      - RTCP (RTP Control Protocol) propagates control information in the opposite direction
      - RTCP is encapsulated in UDP

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

UDP Spoofing

- **Basically like TCP (IP) spoofing**



Intruder

Spoofed UDP Request

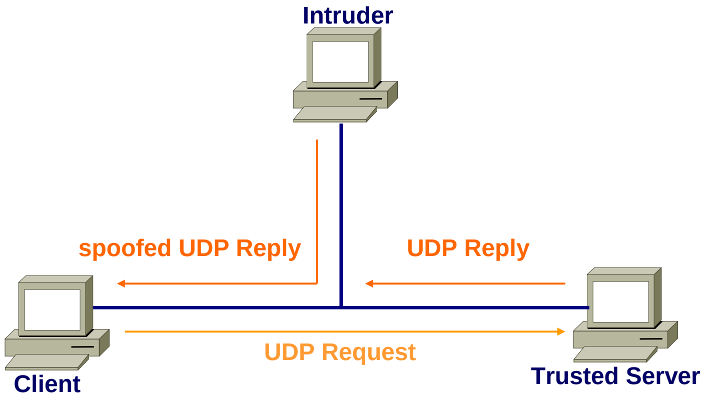
UDP reply from Server

Trusted Client

Server

UDP Hijacking

- Racing against the legitimate server

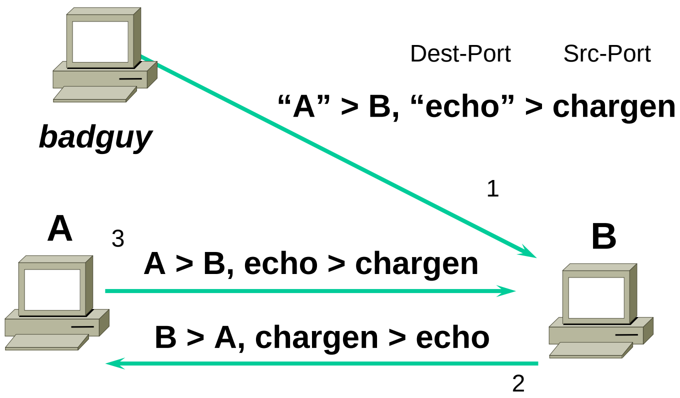


UDP Storm / UDP Bomb

- Intruder sends a spoofed UDP datagram (IP address points to host A) to the echo service port 7 of a host B
- The source port is set to the chargen port 19, which sends a continuous chargen data stream
- The reply of the echo service is interpreted by the spoofed victim machine A as a request by the chargen service
- The reply of the chargen service is interpreted by host B as another request of the echo service
- …..

UDP Storm / UDP Bomb

- One spoofed packet can lock up two computers and the network segment
- It uses simple services -> echo and chargen



Dest-Port Src-Port

“A” > B, “echo” > chargen

badguy

1

A

3

A > B, echo > chargen

B

B > A, chargen > echo

2

UDP Portscan

- Used to determine which UDP services are available
- A zero-length UDP packet is sent to each port
- If an ICMP error message „port unreachable“ occurs the service is assumed to be unavailable
- This type of scan can be slow, because many TCP/IP stack implementations have a limit on the error message rate

UDP Portscan

- **How to do a UDP portscan?**
  - By hand (with packet filter and RAW-socket)
  - Use netcat (http://netcat.sourceforge.net/) and tcpdump
  - Use e.g. nmap –sU <address> (http://www.insecure.org/nmap/)
  - Use online services to test, if your computer is secure (http://www.port-scan.de/)

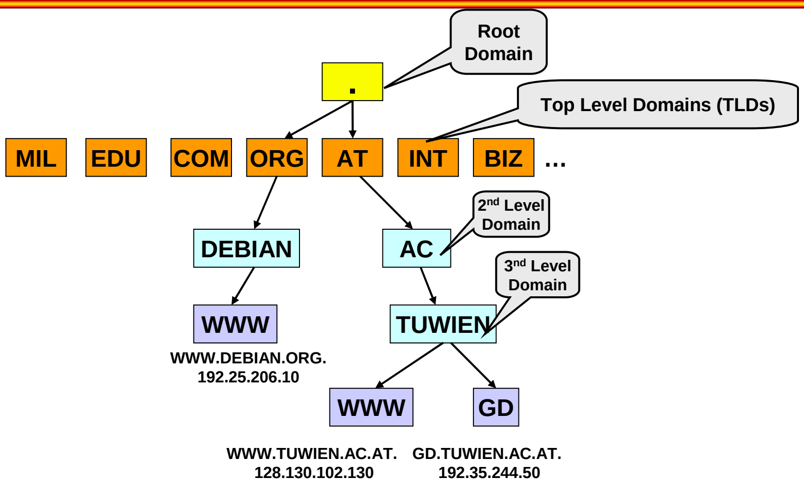
Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

What Basically Does DNS ?

- **DNS "replaces" the IP address of hosts to a human readable format**
  - DNS enables a mapping between names and addresses
  - often called "hostname resolution"
  - due to its size DNS is a world-wide *distributed* database
- **DNS assigns hosts to a tree-like directory hierarchy**
  - each part of the hierarchy is called a "domain“, each hierarchy level is assigned a label, called "domain name"
  - the Domain Name Tree does NOT reflect the physical network structure !!!

Tree of Names


Root Domain
.
Top Level Domains (TLDs)
MIL
EDU
COM
ORG
AT
INT
BIZ
...
2nd Level Domain
DEBIAN
AC
3rd Level Domain
WWW
TUWIEN
WWW.DEBIAN.ORG.
192.25.206.10
WWW
GD
WWW.TUWIEN.AC.AT.
128.130.102.130
GD.TUWIEN.AC.AT.
192.35.244.50

Compare this DNS tree with a file directory tree of a common Operating Systems where **C:\at\ac\tuwien\www\ip_address.txt** is used to specify the location of the file ip_address.txt on the hard disk

Name Servers - DNS Resolver

- **the DNS tree is realized by**
  - Name Servers
- **each Name Server take cares**
  - for a subset of the DNS tree
  - so called "zones"
- **the physical location of name server**
  - has nothing to do with the DNS tree
- **if an IP host wants to resolve a symbolic name**
  - resolver software acting as DNS client will ask a DNS name server using the DNS protocol
  - IP address of name server either manually configured or known through DHCP or explicitly specified by the user

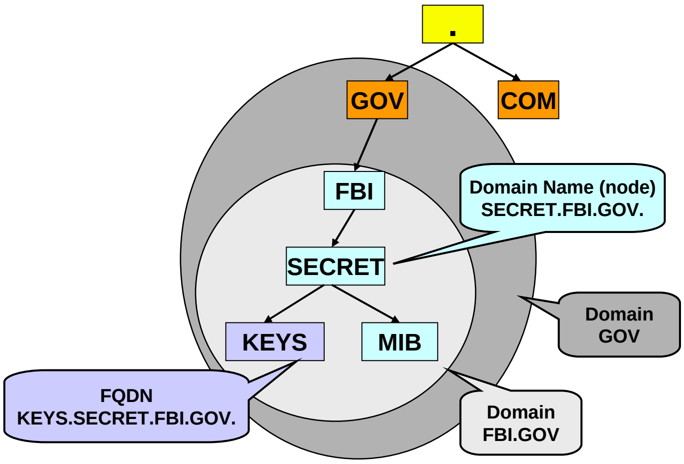
Conventions (1)

- Terminology**: a "Domain" ...**
  - is a complete sub-tree
    - everything under a particular point in the tree
  - relates to the naming structure itself, not the way things are distributed
- Terminology**: a "Domain Name" ...**
  - is the name of a node in the tree (domain, host, ...)
  - consists of all concatenated labels from the root to the current domain, listed from right to left, separated by dots
    - max 255 characters

Conventions (2)

- Terminology**: a "Label" ...**
  - is a component of the domain name
  - need only be unique at a particular point in the tree
    - that is, both "name.y.z" and "name.x.y.z" are allowed
    - max 63 characters
  - DNS is not case sensitive !
    - "www.nic.org" is the same as "WWW.NIC.ORG"
  - due to SMTP restrictions, domain names may contain only characters of {a-z, A-Z, 0-9, "-"}
    - there are some new conventions concerning national characters
- Terminology**: a "Fully Qualified Domain Name" (FQDN)**
  - concatenation of all labels of including trailing dot **". "**

Example for Terminology


.
GOV
COM
FBI
Domain Name (node)
SECRET.FBI.GOV.
SECRET
Domain
GOV
KEYS
MIB
FQDN
KEYS.SECRET.FBI.GOV.
Domain
FBI.GOV

The Root Domain

- **the root of the DNS tree is denoted as a single dot "."**
  - each domain name without this root-dot is only a relative domain name
    - although, most applications do not follow this rule
    - but essential in BIND configuration files (master files)
  - otherwise it is a Fully Qualified Domain Name (FQDN) which exactly identifies a single host from all hosts in the world
- **the root is implemented by *several* root-servers**
  - name server at the highest hierarchy level
- **below the root, a domain may be called top-level, second-level, third-level etc...**

Top Level Domains (RFC1591)

- **inside US: "generic domains"**
  - **com** - Commercial
  - **edu** - Educational
  - **org** - Non Profit Organizations (NPOs)
  - **net** - Networking providers
  - **mil** - US military
  - **gov** - US goverment
  - **int** - International organizations
- **outside US: two letter country code**
  - defined in ISO-3166
  - examples: **uk** (United Kingdom), **fr** (France), **us** (United States), **de** (Germany), **at** (Austria), **ax** (Antarctica)
  - Note: country code does not reflect real location !

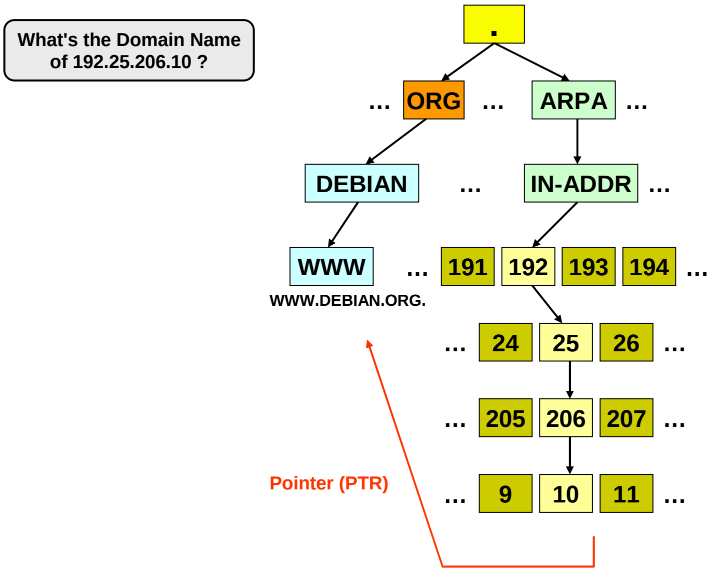
IN-ADDR.ARPA (1)

- **special feature: the *in-addr.arpa domain***
  - used to support gateway location
  - enables reverse lookups: given an IP-address the associated hostname can be found
- **without the IN-ADDR.ARPA domain**
  - an *exhaustive search* in the domain space would be necessary to find any desired hostname
- **commonly used by**
  - WWW servers to log its users in a file
  - IRC servers that want to restrict their service inside a certain domain
    - E.g. a closed chat/discussion group exclusive for domains under IEEE.ORG

IN-ADDR.ARPA (2)

- **the domain in-addr.arpa is structured according to the IP address**
  - this special domain begins at "IN-ADDR.ARPA"
  - its substructure follows the Internet addressing structure
- **each domain name has up to 4 additional labels**
  - each label represents one octet of the IP address
    - expressed as character string for its decimal value ("0" - "255")
    - the reverse host/domain names are organized on byte boundaries
  - Note: labels are attached to the suffix in reverse order
    - e.g. data for internet address 216.32.74.50 is found at 50.74.32.216.IN-ADDR.ARPA
    - hosts have all four labels specified

IN-ADDR.ARPA (3)


What's the Domain Name of 192.25.206.10 ?
.
... ORG ... ARPA ...
DEBIAN ... IN-ADDR ...
WWW ... 191 192 193 194 ...
WWW.DEBIAN.ORG.
... 24 25 26 ...
... 205 206 207 ...
Pointer (PTR)
... 9 10 11 ...

What is BIND ?

- **the Berkeley Internet Name Domain (BIND)**
  - implemented by Paul Vixie as an Internet name server for BSD-derived systems
  - most widely used name server on the Internet
  - version numbers: 4 (old but still used), 8 and 9 (new)
- **BIND consists of**
  - a name server called named ("d" stands for "daemon")
  - a resolver library for client applications
    - The "resolver" is a collection of functions like gethostbyname(2) and gethostbyaddr(2)
- **technically, BIND and DNS deal primarily with zones**
  - a zone is a part of the domain space

What is a Zone ?

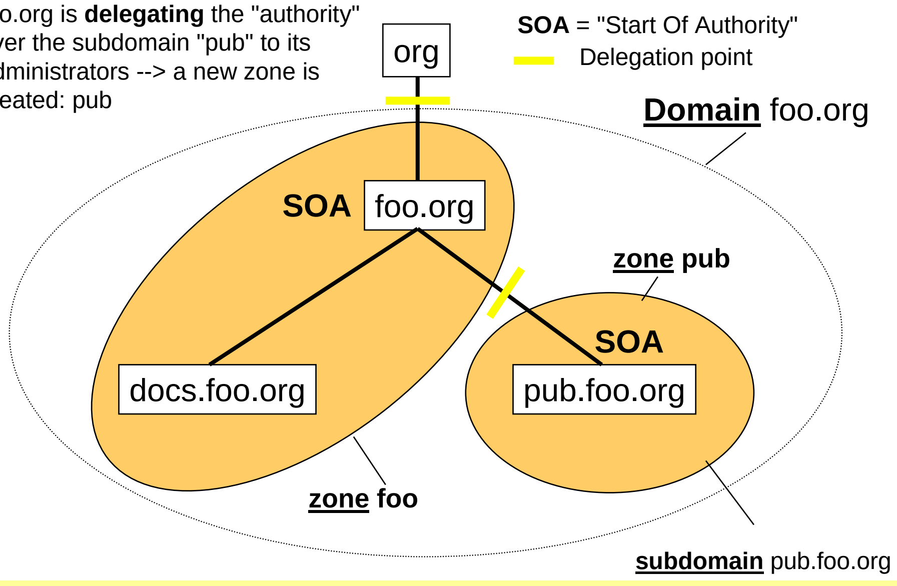
- **a zone is a "point of delegation"**
  - contains all names from this point downwards the domain-tree except those which are delegated to other zones (to other name servers)
  - a zone can span over a whole domain or just be part of it
- ***in other words:* a zone is a pruned domain !**
  - pruning occurs when zones are delegated
  - zones relate to the way the database is partitioned and distributed
- **a name server is *authoritative* over a domain**
  - if he keeps a *master file* (zone file) with information of that domain

Zones and SOA

foo.org is **delegating** the "authority" over the subdomain "pub" to its administrators --> a new zone is created: pub

SOA = "Start Of Authority"

▬ Delegation point

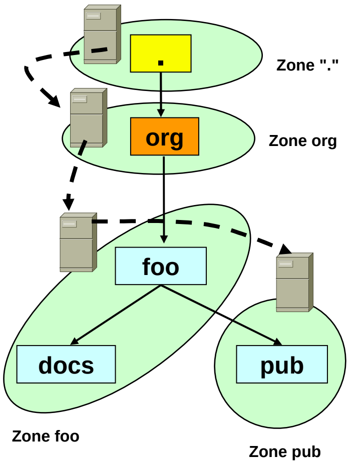


Delegation and Name Servers

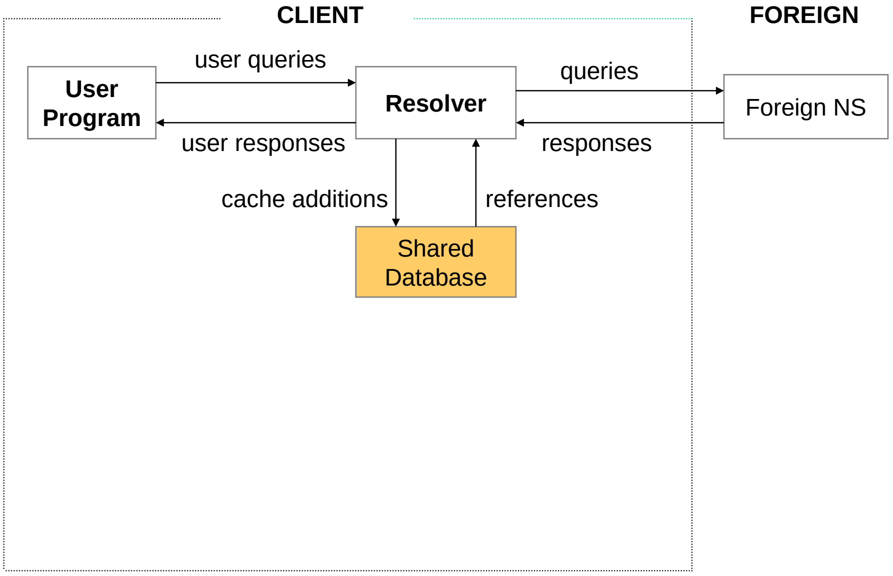
Root Server responsible for root domain delegates authority for building symbols org. to NS below which holds the master file for zone org

NS responsible for domain org delegates authority for building symbols foo.org. to NS below which holds the master file for zone foo

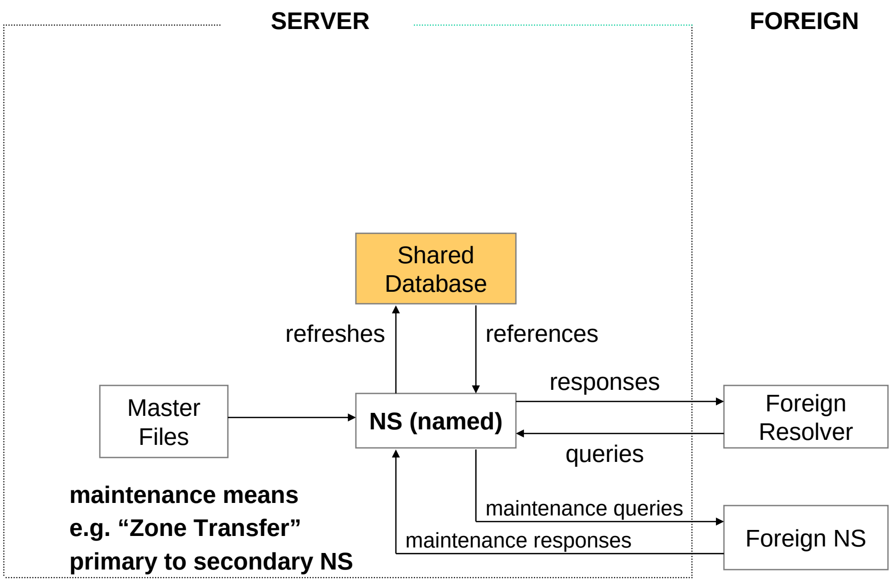
NS responsible for domain foo.org delegates authority for building symbols pub.foo.org. to NS below which holds the master file for zone pub


.
Zone "."
org
Zone org
foo
docs
pub
Zone foo
Zone pub

BIND Principles (Client)

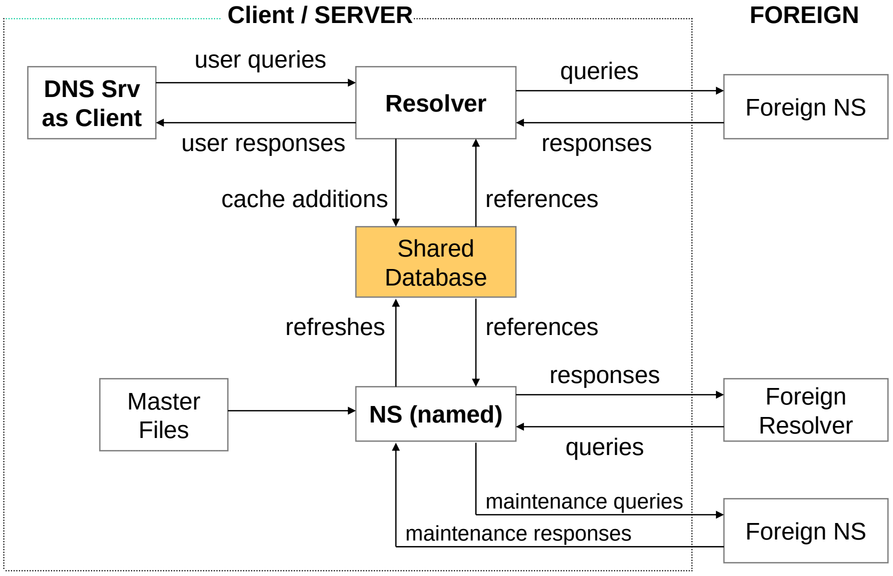

CLIENT
FOREIGN
User Program
user queries
user responses
Resolver
queries
responses
Foreign NS
cache additions
references
Shared Database

BIND Principles (Server)


SERVER
FOREIGN
Shared Database
refreshes
references
responses
Master Files
NS (named)
Foreign Resolver
queries
maintenance queries
maintenance responses
Foreign NS

maintenance means e.g. “Zone Transfer” primary to secondary NS

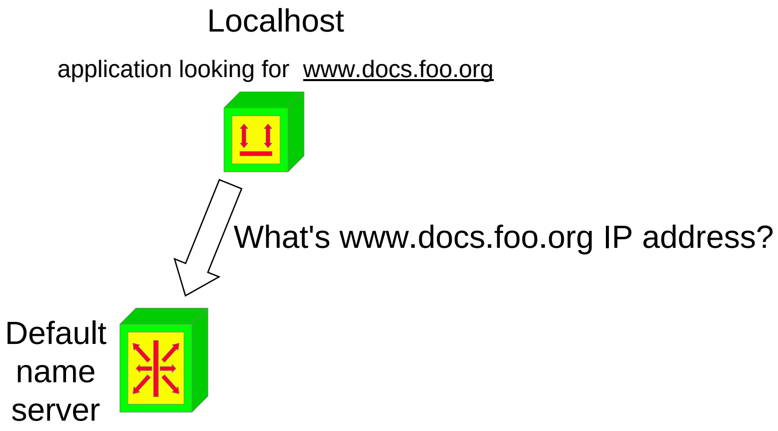
BIND Principles (Server complete)


Client / SERVER
FOREIGN
user queries
DNS Srv as Client
Resolver
queries
Foreign NS
user responses
responses
cache additions
references
Shared Database
refreshes
references
responses
Master Files
NS (named)
Foreign Resolver
queries
maintenance queries
maintenance responses
Foreign NS

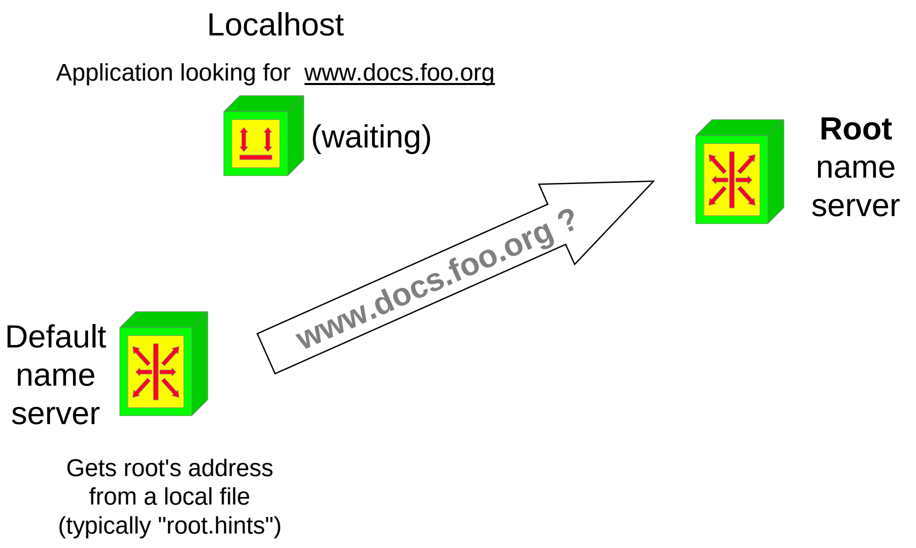
BIND Principles

- **applications running on a client use the resolver to send *name resolution queries* to a name server**
  - each client-host requires a preconfigured IP address of one (or several) *default name server(s)*
- **a name server responses to this query after retrieving the requested data either**
  - by recursive queries -> the job is forwarded
  - by iterative queries -> the NS replies with a list of authoritative NS's to be queried by the client
  - from its cache -> the NS supplies non-authoritative data
  - or by its own zone data contained in its master file:
    - the NS is authoritative for that requested zone

Recursive Query (1)


Localhost
application looking for www.docs.foo.org
What's www.docs.foo.org IP address?
Default name server

Iterative Queries (2)



Localhost

Application looking for www.docs.foo.org

(waiting)

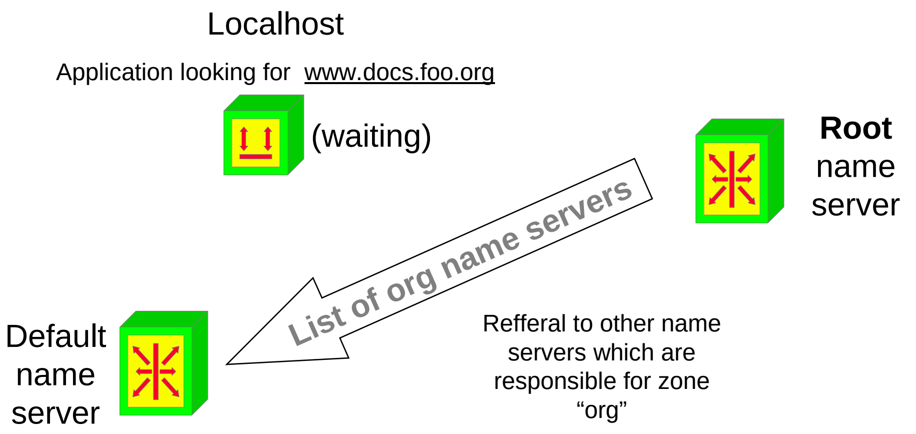
Root name server

www.docs.foo.org ?

Default name server

Gets root's address from a local file (typically "root.hints")

Iterative Queries (3)



Localhost

Application looking for www.docs.foo.org

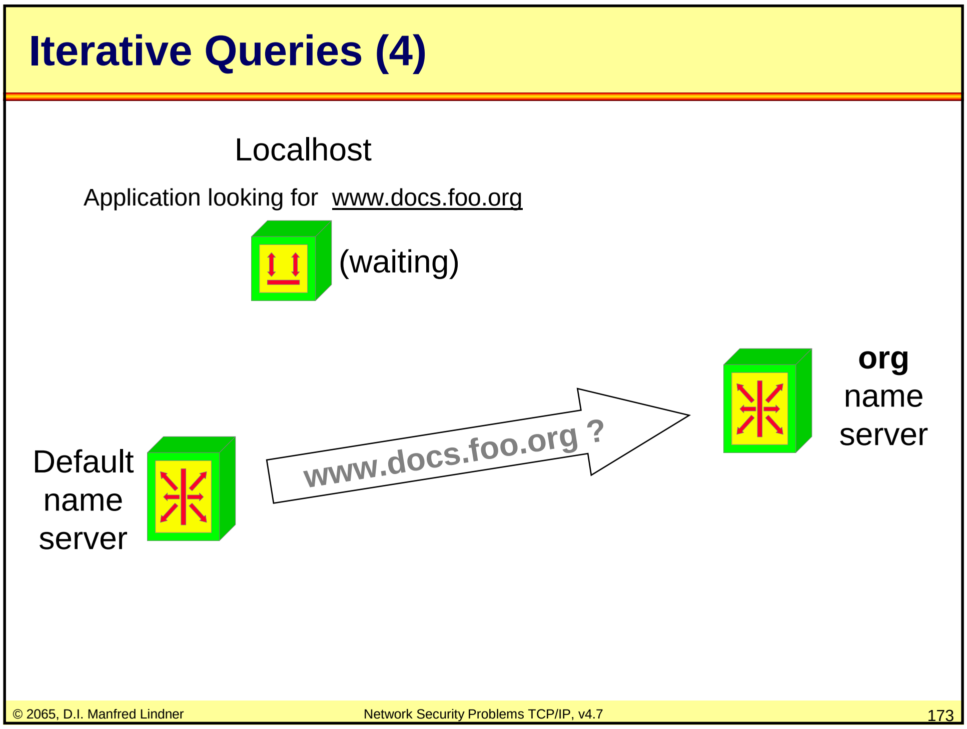
(waiting)

Root name server

List of org name servers

Default name server

Refferal to other name servers which are responsible for zone “org”



Iterative Queries (4)

Localhost

Application looking for www.docs.foo.org

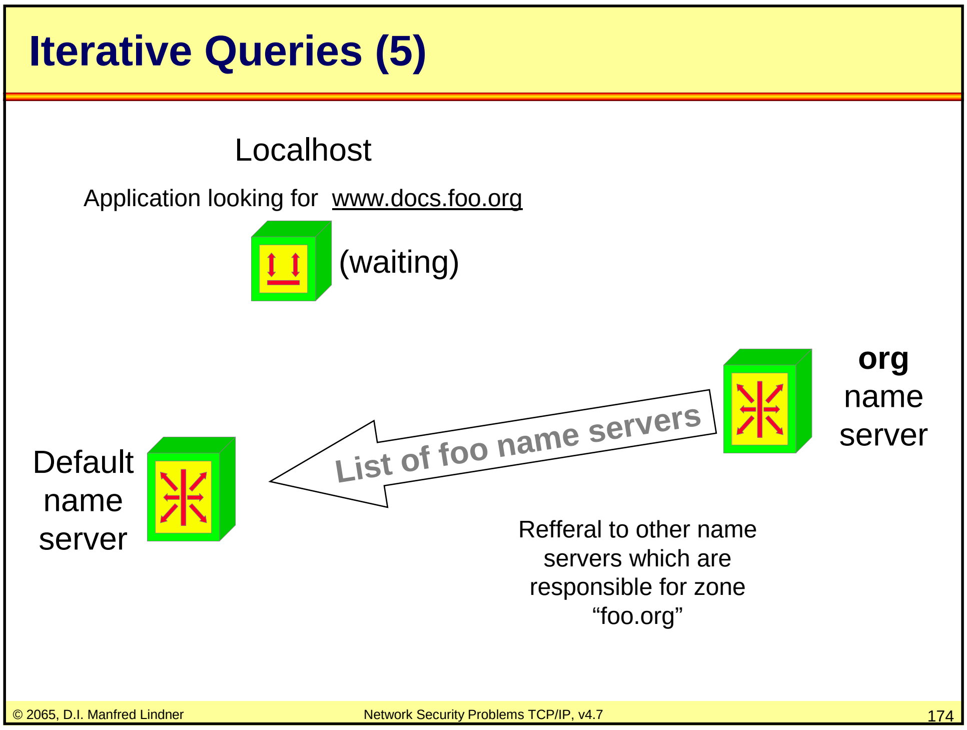
(waiting)

Default name server

www.docs.foo.org ?

org name server

© 2065, D.I. Manfred Lindner
Network Security Problems TCP/IP, v4.7
173



Iterative Queries (5)

Localhost

Application looking for www.docs.foo.org

(waiting)

Default name server

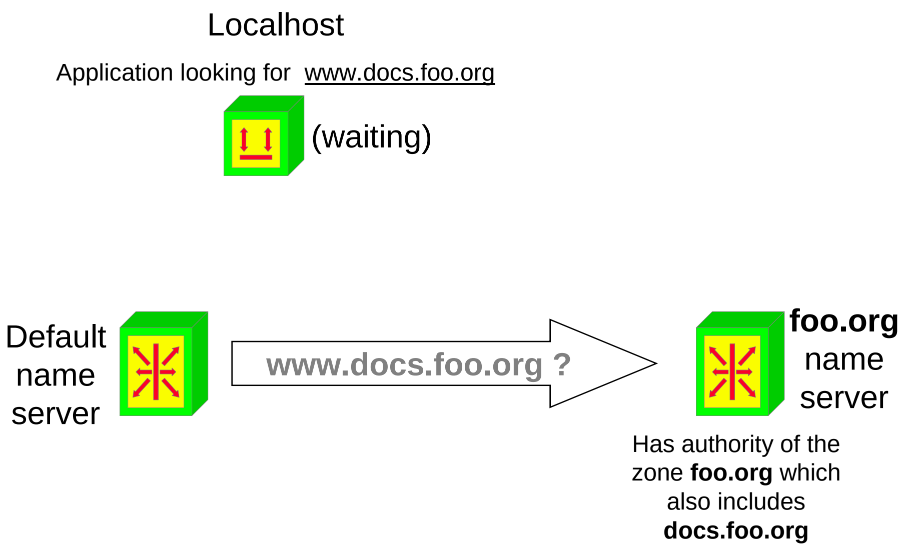
List of foo name servers

org name server

Refferal to other name servers which are responsible for zone “foo.org”

© 2065, D.I. Manfred Lindner
Network Security Problems TCP/IP, v4.7
174

Iterative Queries (6)



Localhost

Application looking for www.docs.foo.org

(waiting)

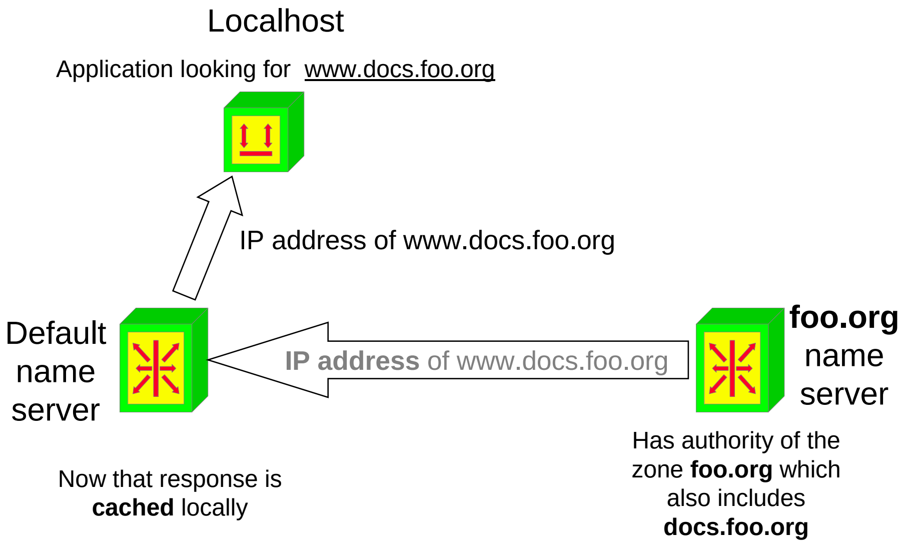
Default name server

www.docs.foo.org ?

foo.org name server

Has authority of the zone **foo.org** which also includes **docs.foo.org**

Iterative Queries (7)



Localhost

Application looking for www.docs.foo.org

IP address of www.docs.foo.org

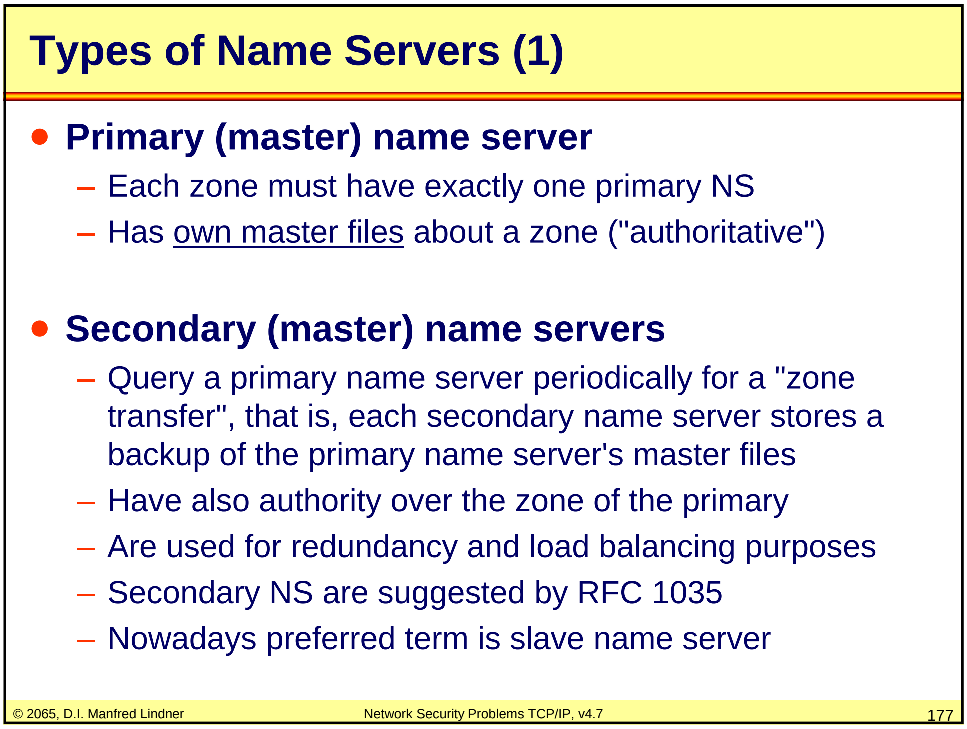
Default name server

IP address of www.docs.foo.org

foo.org name server

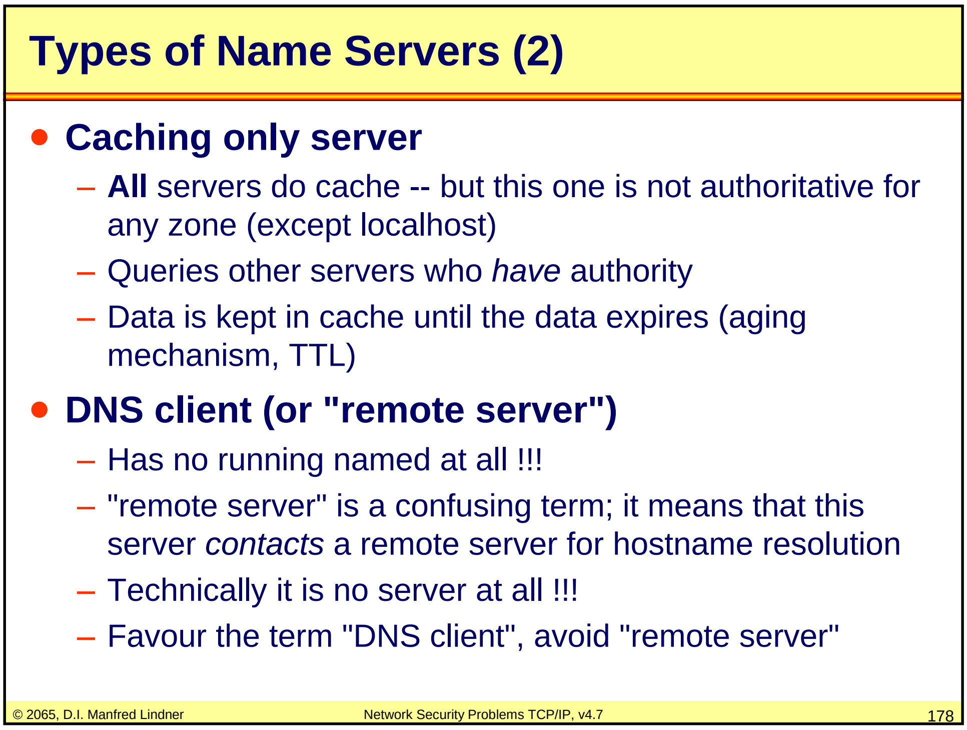
Now that response is **cached** locally

Has authority of the zone **foo.org** which also includes **docs.foo.org**



Types of Name Servers (1)

- **Primary (master) name server**
  - Each zone must have exactly one primary NS
  - Has own master files about a zone ("authoritative")
- **Secondary (master) name servers**
  - Query a primary name server periodically for a "zone transfer", that is, each secondary name server stores a backup of the primary name server's master files
  - Have also authority over the zone of the primary
  - Are used for redundancy and load balancing purposes
  - Secondary NS are suggested by RFC 1035
  - Nowadays preferred term is slave name server



Types of Name Servers (2)

- **Caching only server**
  - **All** servers do cache -- but this one is not authoritative for any zone (except localhost)
  - Queries other servers who *have* authority
  - Data is kept in cache until the data expires (aging mechanism, TTL)
- **DNS client (or "remote server")**
  - Has no running named at all !!!
  - "remote server" is a confusing term; it means that this server *contacts* a remote server for hostname resolution
  - Technically it is no server at all !!!
  - Favour the term "DNS client", avoid "remote server"

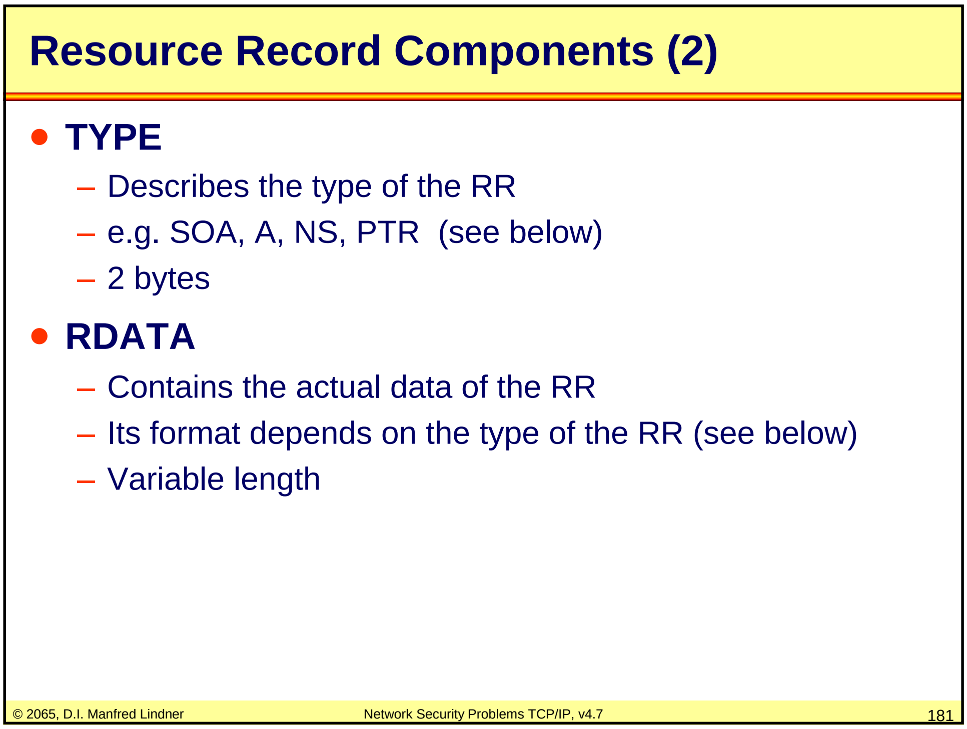
Resource Records

- **All data contained in a master file is split up into Resource Records (RRs)**
- **All DNS operations are formulated in terms of RRs (RFC 1035)**
  - Each query is answered with a copy of matching RRs !!!
  - RRs are the smallest unit of information available through DNS
- **RR format**
  - 5 fields, separated by spaces or tabs:

[DOMAIN] [TTL] [CLASS] TYPE RDATA

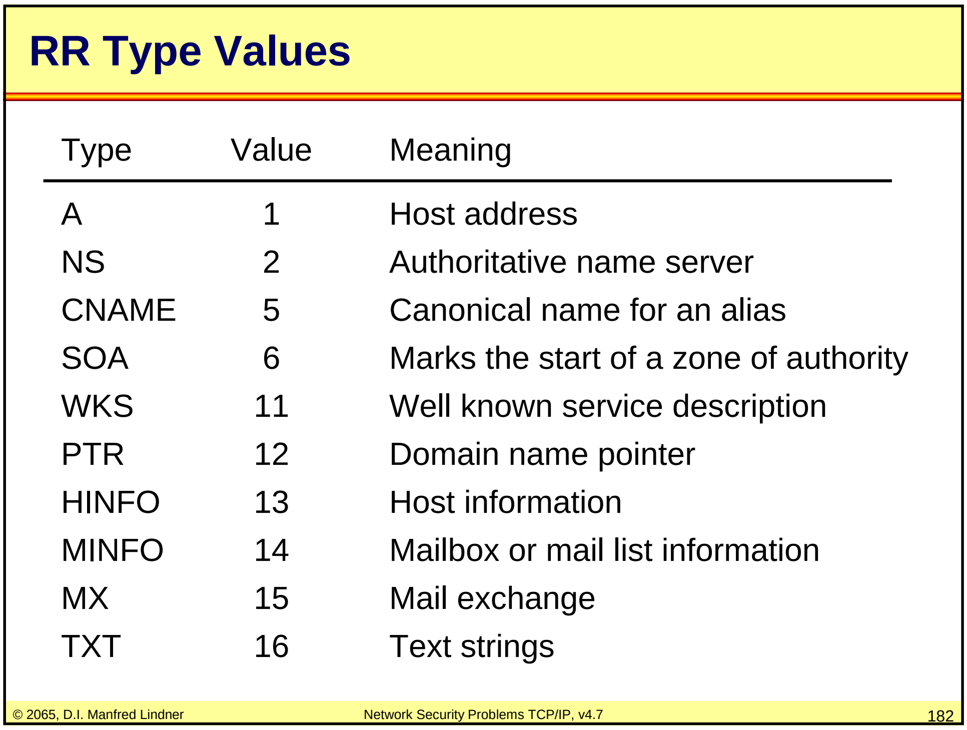
Resource Record Components (1)

- **DOMAIN**
  - Domain name to which the entry applies
  - If no domain name is given the RR applies to the domain of the previous RR
- **TTL**
  - Time To Live = time in seconds this RR is valid after it has been retrieved from the server
  - 8 digit decimal number
- **CLASS**
  - Address class: IN for Internet, CH for CHAOS, HS for Hesiod (MIT)
  - 2 bytes



Resource Record Components (2)

- **TYPE**
  - Describes the type of the RR
  - e.g. SOA, A, NS, PTR (see below)
  - 2 bytes
- **RDATA**
  - Contains the actual data of the RR
  - Its format depends on the type of the RR (see below)
  - Variable length



RR Type Values

Type	Value	Meaning
A	1	Host address
NS	2	Authoritative name server
CNAME	5	Canonical name for an alias
SOA	6	Marks the start of a zone of authority
WKS	11	Well known service description
PTR	12	Domain name pointer
HINFO	13	Host information
MINFO	14	Mailbox or mail list information
MX	15	Mail exchange
TXT	16	Text strings

Types of Resource Records (1)

- **SOA - Start of Authority RR**
  - Marks the beginning of a zone; typically seen as the first record in a master file
  - All records following the SOA RR contain authoritative information for the domain
  - Every master file included by a primary statement must contain an SOA record for this zone

SOA RDATA fields:

- MNAME (or "ORIGIN")
  - Canonical hostname of the primary server for this domain
  - Usually given as absolute name (FQDN)

SOA RDATA fields cont.

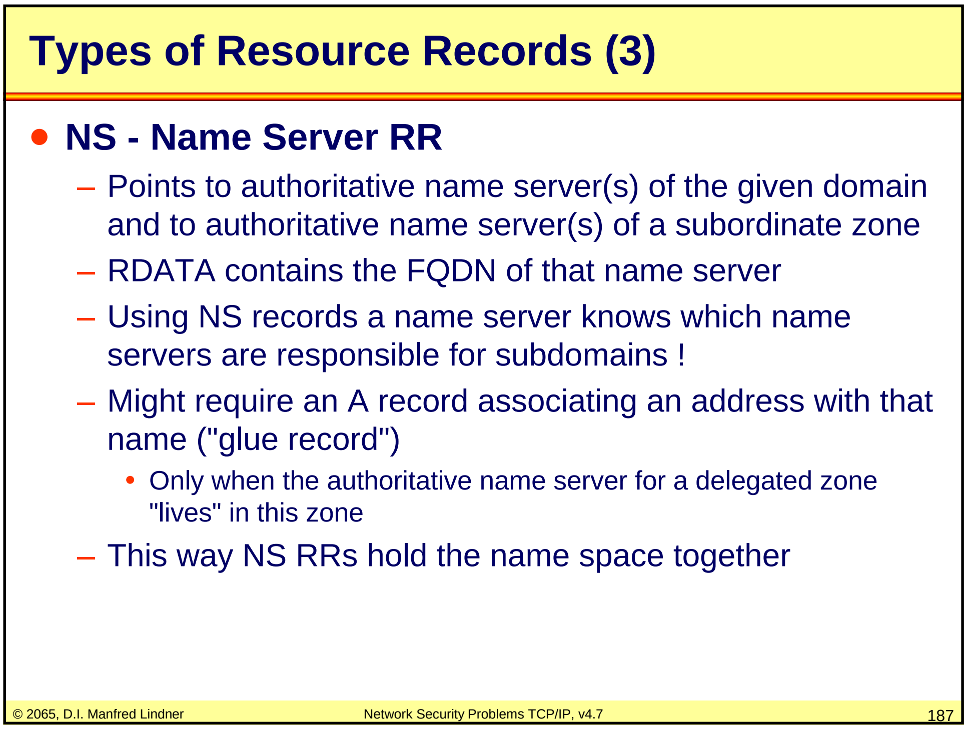
- RNAME (or "CONTACT")
  - E-Mail address of an administrator responsible for this domain
  - The "@" character must be replaced with a dot
- SERIAL
  - Version number of the zone file
  - Is used by secondary name servers to recognize changes of the zone file
  - Should be incremented when changes are applied to the zone
- REFRESH
  - 32 bit time interval in seconds that a secondary name server should wait between checking this SOA record
- RETRY
  - 32 bit time value in seconds that should elapse before a failed refresh should be retried by a secondary name server

SOA RDATA fields cont.

- EXPIRE
  - 32 bit time value in seconds after which this zone data should not be regarded as authoritative any longer
  - After this time a server may discard all zone data
  - Normally a very large period, e.g. 42 days
- MINIMUM
  - Minimum 32 bit TTL value in seconds
  - Is a lower bound on the TTL field for all RRs in a zone
  - Only used for normal responses (not zone transfers)

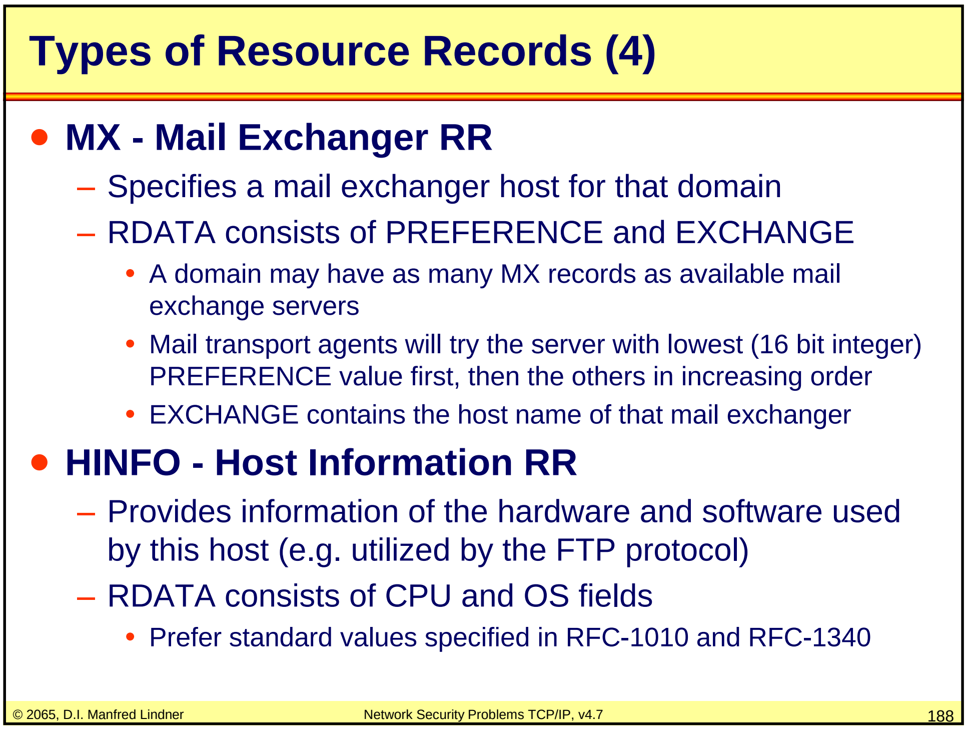
Types of Resource Records (2)

- **A - Address RR**
  - Most important -- associates an IP address with one canonical hostname
  - RDATA consists of a 32-bit IP address
  - Each host can have exactly as many A records as it has network interfaces
- **CNAME - Canonical Name RR**
  - Is like an alias or a symbolic link to a canonical hostname
  - RDATA contains the canonical name
- **PTR - POINTER RR**
  - Points to another location in the domain name space
  - RDATA contains the domain name



Types of Resource Records (3)

- **NS - Name Server RR**
  - Points to authoritative name server(s) of the given domain and to authoritative name server(s) of a subordinate zone
  - RDATA contains the FQDN of that name server
  - Using NS records a name server knows which name servers are responsible for subdomains !
  - Might require an A record associating an address with that name ("glue record")
    - Only when the authoritative name server for a delegated zone "lives" in this zone
  - This way NS RRs hold the name space together



Types of Resource Records (4)

- **MX - Mail Exchanger RR**
  - Specifies a mail exchanger host for that domain
  - RDATA consists of PREFERENCE and EXCHANGE
    - A domain may have as many MX records as available mail exchange servers
    - Mail transport agents will try the server with lowest (16 bit integer) PREFERENCE value first, then the others in increasing order
    - EXCHANGE contains the host name of that mail exchanger
- **HINFO - Host Information RR**
  - Provides information of the hardware and software used by this host (e.g. utilized by the FTP protocol)
  - RDATA consists of CPU and OS fields
    - Prefer standard values specified in RFC-1010 and RFC-1340

Types of Resource Records (5)

- **WKS - Well Known Service RR**
  - Specifies a well known service supported by a particular protocol on a particular host
  - RDATA contains
    - ADDRESS (32 bit) IP Address
    - PROTOCOL (8 bit) IP protocol number
    - BIT MAP (variable length) indicates the TCP port number, e.g. the 26th bit set indicates port 25 - SMTP
- **LOC - Location (EXPERIMENTAL)**
  - Allows DNS to carry location information about hosts and networks (example application: xtraceroute)
  - RDATA contains latitude, longitude and altitude information fields

The "DNS Protocol"

- **DNS messages utilize TCP or UDP as transport protocol**
  - UDP for standard queries (need for performance)
  - TCP for zone transfers (need for reliability)
- **Well known port number 53 (server side)**
- **DNS messages using UDP are restricted to 512 bytes**
  - Longer messages are truncated and the TC bit is set in the header

Message Format

DNS messages have always the following 5 sections:

HEADER	Specifies which sections are present, query or response, etc
QUESTION	Contains the question for the NS
ANSWER	Contains **RRs** answering the question
AUTHORITY	Contains **RRs** pointing toward an authority
ADDITIONAL	Contains **RRs** holding additional information

Some sections (except HEADER) may be <u>empty</u> in certain cases

Header Section

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="16">IDENTIFICATION</td></tr>
<tr><td>QR</td><td colspan="4">OPCODE</td><td>AA</td><td>TC</td><td>RD</td><td>RA</td><td colspan="3">Z</td><td colspan="4">RCODE</td></tr>
<tr><td colspan="16">QDCOUNT (number of questions)</td></tr>
<tr><td colspan="16">ANCOUNT (number of answers)</td></tr>
<tr><td colspan="16">NSCOUNT (number of authority)</td></tr>
<tr><td colspan="16">ARCOUNT (number of additional)</td></tr>
</table>

Header Fields (1)

- **IDENTIFICATION**
  - 16 bit identifier assigned by the requesting program
  - the corresponding reply gets the same identifier
- **QR**
  - query = 0, response = 1
- **OPCODE**
  - Specifies the kind of query in this message

    0 .... standard query (QUERY)
    1 .... inverse query (IQUERY); IN-ADDR.ARPA !!!
    2 .... server status request (STATUS)
    3 -15 ... reserved

Header Fields (2)

- **AA**
  - Authoritative Answer
  - The responding NS is an authority for the domain name in the question section
  - If set, the data comes directly from a primary or secondary name server and not from a cache
- **TC**
  - TrunCation
  - Indicates that this message has been truncated (due to transmission channel's max message size)
- **RD**
  - Recursion Desired
  - The NS should solve the query recursively

Header Fields (3)

- **RA**
  - Recursion Available
  - May be set or cleared in a response
  - Indicates whether recursive queries are supported by the NS
- **Z**
  - Reserved
  - Must be zero

Header Fields (4)

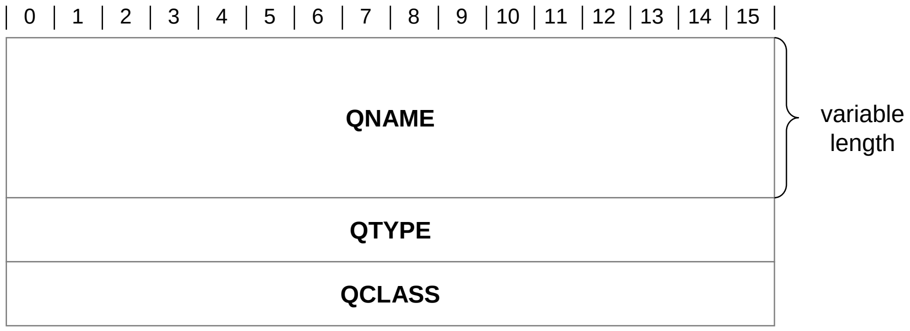
- **RCODE**
  - Response Code
  - 0 ... *no error*
  - 1.... *format error* - the NS was not able to interpret the query
  - 2 ... *server failure* - the NS has problems
  - 3 ... *name error* - an authoritative NS signals that the requested domain does not exist
  - 4 ... *not implemented* - the NS does not support this kind of query
  - 5 ... *refused* - the NS refuses the required operation for policy reasons
  - 6-15 ... reserved for future use

Header Fields (5)

- **QDCOUNT**
  - Specifies the number of entries in the question section
- **ANCOUNT**
  - Specifies the number of RRs in the answer section
- **NSCOUNT**
  - Specifies the number of NS RRs in the authority records section
- **ARCOUNT**
  - Specifies the number of RRs in the additional records section

Question Section

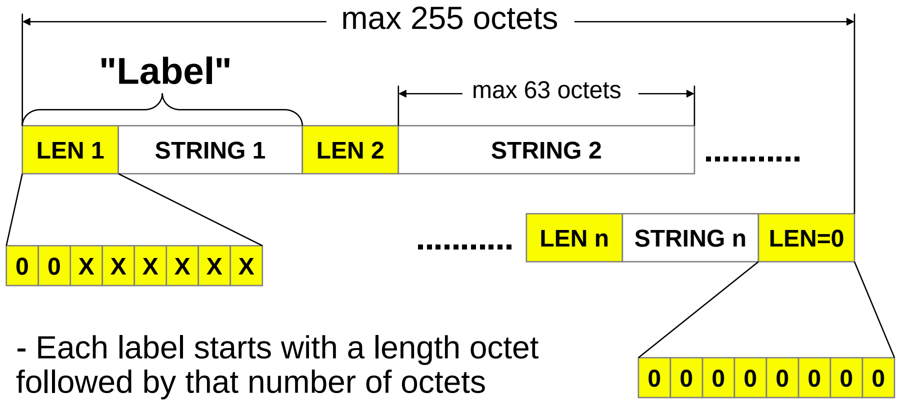
The question section contains QDCOUNT entries, each of the following format:



Question Fields

- **QNAME**
  - A domain name represented as a set of labels
    See the domain name message format below
  - Can have an odd number of octets, no padding is used as reminder
- **QTYPE**
  - Type of query; values are a superset of the TYPE values in RRs
    - AXFR (252) request for a transfer of the entire zone
    - " * " (255) request for all records
- **QCLASS**
  - Class of the query; values are a superset of the CLASS values in RRs (usually "IN" for Internet, " * " for any class)

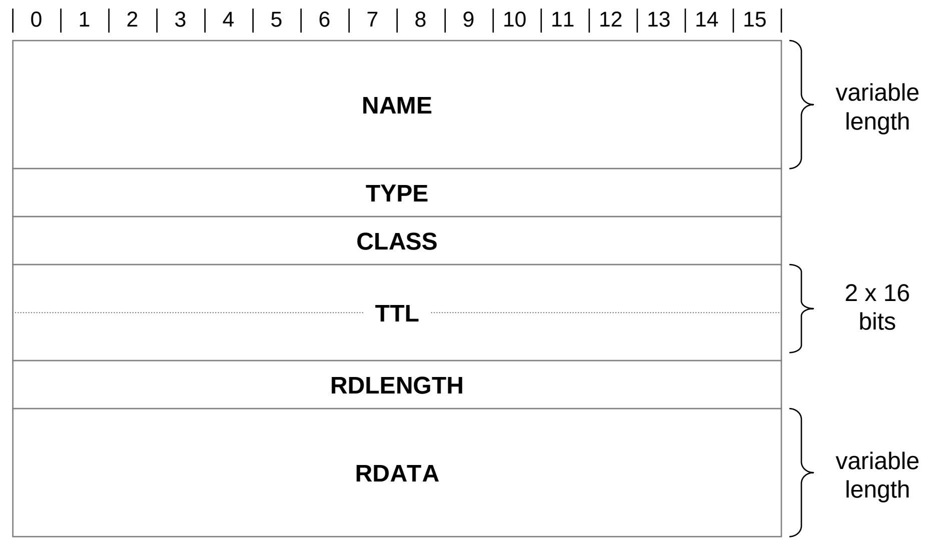
Domain Names in Messages


max 255 octets
"Label"
max 63 octets
LEN 1
STRING 1
LEN 2
STRING 2
....
0 0 X X X X X X
....
LEN n
STRING n
LEN=0
0 0 0 0 0 0 0 0

- Each label starts with a length octet followed by that number of octets

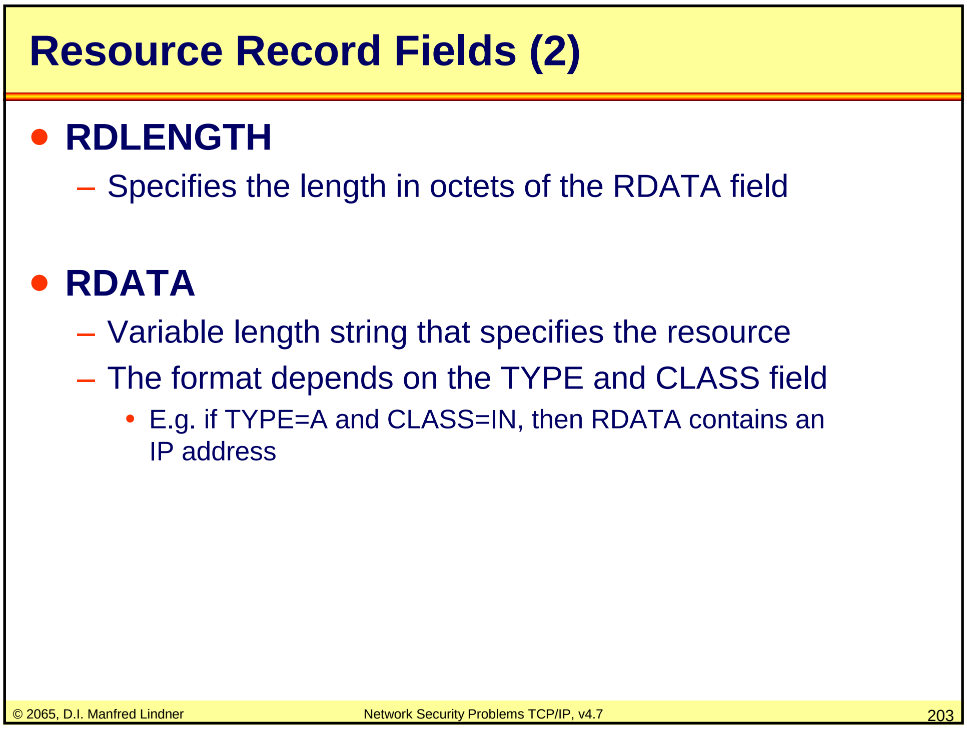
- The domain name is terminated with a zero length octet (= "null label" for the root)

Resource Record Format in Answers, Authorative and Additional Fields


| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
NAME
variable length
TYPE
CLASS
TTL
2 x 16 bits
RDLENGTH
RDATA
variable length

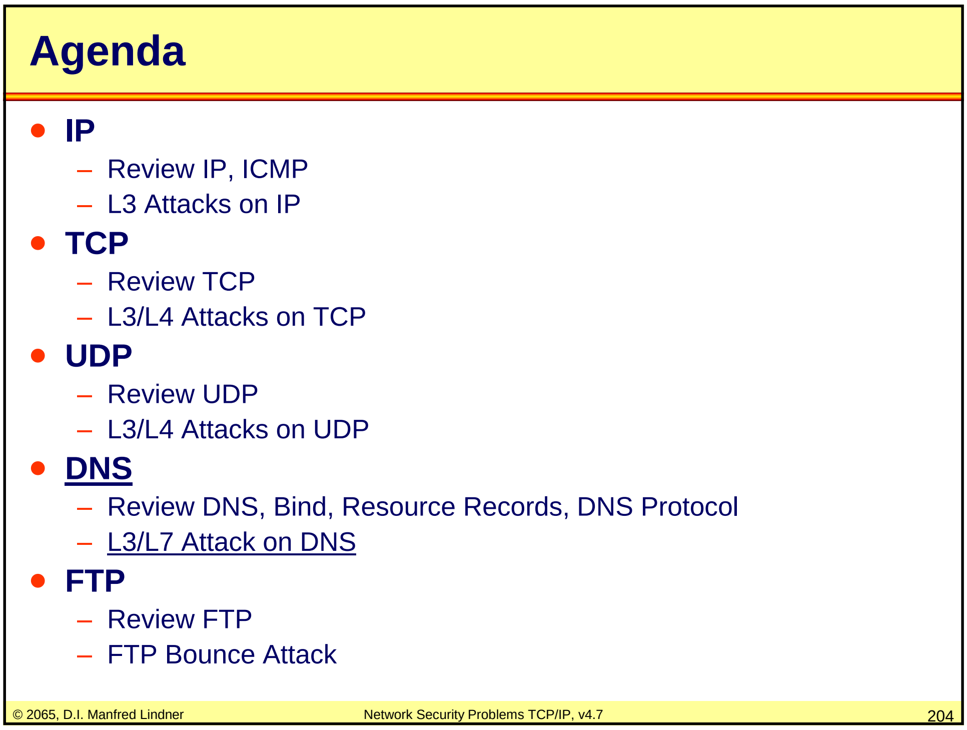
Resource Record Fields (1)

- **NAME**
  - Domain name to which this RR refers
- **TYPE**
  - Specifies the meaning of the data in the RDATA field
  - e.g. A, CNAME, NS, SOA, PTR, ...
- **CLASS**
  - Specifies the class of the data in the RDATA field
- **TTL**
  - Specifies the duration this RR may be cached before it should be discarded
  - Zero values suggest that this RR should not be cached
  - 32 bit, time in seconds



Resource Record Fields (2)

- **RDLENGTH**
  - Specifies the length in octets of the RDATA field
- **RDATA**
  - Variable length string that specifies the resource
  - The format depends on the TYPE and CLASS field
    - E.g. if TYPE=A and CLASS=IN, then RDATA contains an IP address



Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

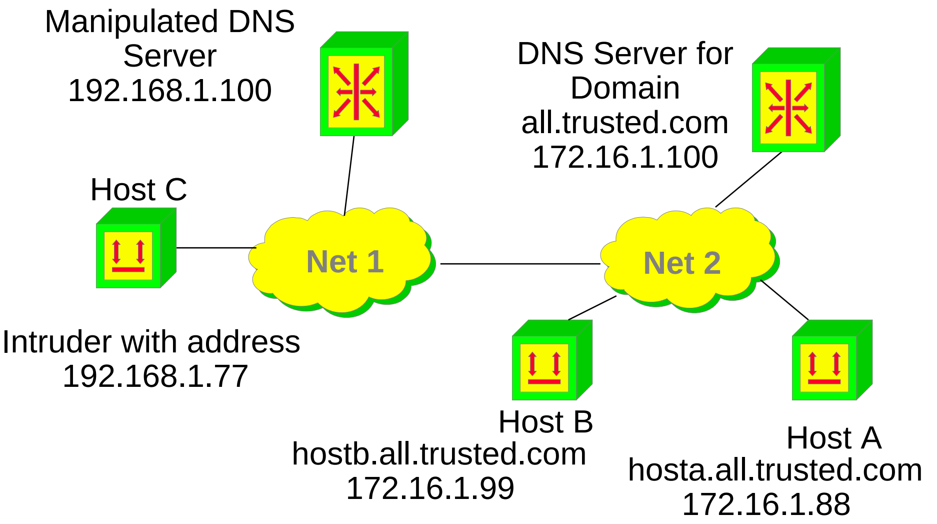
DNS Attacks

- **DNS may provide too much information**
  - HINFO records (Info about OS of a host)
  - WKS (well known service) records
  - Zone transfers (query for entire content of a zone)
  - Scanning is not necessary in most cases
- **Mostly UDP is used, so it is vulnerable to spoofing and hijacking**
  - “DNS Spoofing”
  - “DNS Hijacking”
- **Also vulnerable to DNS cache poisoning attacks**
  - DNS server cache, Client browser cache
  - “DNS Cache Poisoning”

Example for DNS Spoofing 1

- **Sometime authentication is performed with the DNS name**
  - E.g. hostb.all.trusted.com may login using rlogin without specifying a username/ password pair
- **Concept**
  - In-addr.arpa query is used
    - Inverse lookup to get the name of the machine
  - A DNS query is forwarded to another authoritative DNS server (under control of the attacker)
  - This DNS server replies with a faked resource record for the asked IP address
    - A spoofed DNS name will be returned

Example for DNS Spoofing 2



Manipulated DNS
Server
192.168.1.100

DNS Server for
Domain
all.trusted.com
172.16.1.100

Host C

Net 1

Net 2

Intruder with address
192.168.1.77

Host B
hostb.all.trusted.com
172.16.1.99

Host A
hosta.all.trusted.com
172.16.1.88

Example for DNS Spoofing 3

- Host C (192.168.1.77) opens a TCP connection to Server A (172.16.1.88)
- Server A asks its DNS server (172.16.1.100) to look up the name
  - Inverse Lookup
- A‘ s DNS server can‘ t resolve this address and forwards the query to C´s DNS server
- C‘ s DNS server (192.168.1.100) gets the request and injects a reply with a wrong but trusted name (e.g. hostb.all.trusted.com)
- A gets from its DNS server the answer 192.168.1.77 is hostb.all.trusted. com and allows C to log in without username/ password

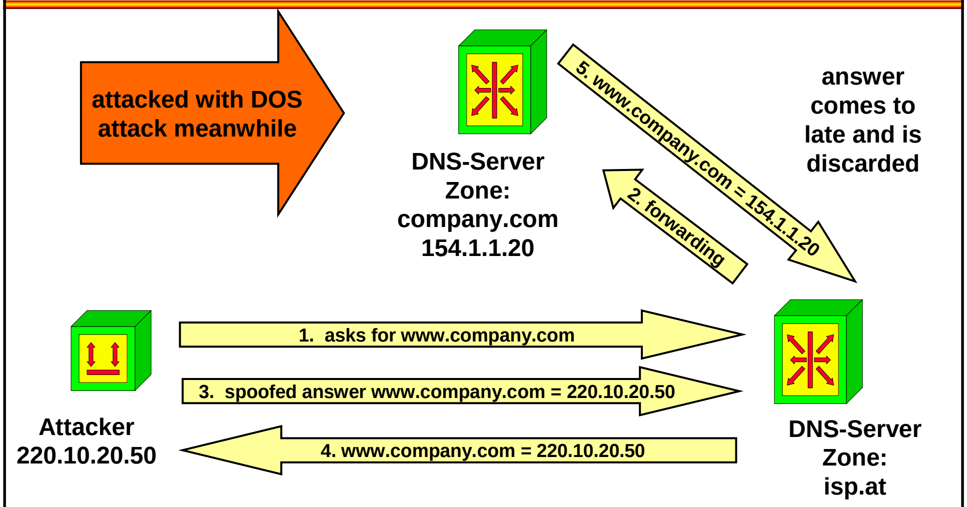
Example for DNS Spoofing 4

- **Can be prevented with**
  - Double reverse lookups
- **Given the IP address C**
  - Host A obtain the name N
- **By using the so obtained name N**
  - Ask (DNS query) for IP address for that name again
- **Check if IP addresses are equal**
  - If yes -> ok
  - If no -> then don’t let the connection to be setup

DNS Hijacking

- **It is possible to perform DNS hijacking by**
  - Racing with the server with respect to a client
  - Racing with a server with respect to another server
- **„Blind“ DNS hijacking**
  - Requires to guess the DNS request ID
    - Many implementations just use sequential numbers for ID´s
  - Blind means that an attacker can not sniff DNS request sent to another server
- **How to guess the DNS request ID**
  - One possibility is to brute-force while keeping real DNS silent by the help of an DoS attack
  - Other possibility: Attacker asks for a domain that is not in the zone of the to be attacked DNS server “victim”
  - DNS server “victim” asks responsible (correct) DNS server, but this server is under the control of the intruder
    - So he can see the numbers used by the “victim” DNS server

DNS Cache Poisoning Method 1



attacked with DOS attack meanwhile

DNS-Server
Zone:
company.com
154.1.1.20

5. www.company.com = 154.1.1.20

answer comes to late and is discarded

2. forwarding

Attacker
220.10.20.50

1. asks for www.company.com

3. spoofed answer www.company.com = 220.10.20.50

4. www.company.com = 220.10.20.50

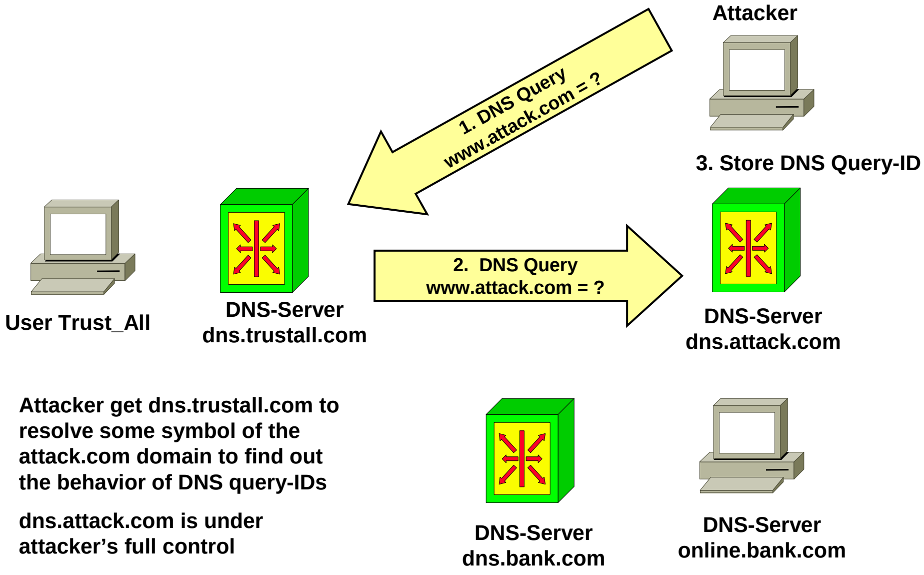
DNS-Server
Zone:
isp.at

In step 3 the request ID of the DNS server used for step 2 needs to be guessed: brute force may be work because only 65536 different IDs are possible → maybe 60s needed to brute force → DoS attack against company.com is performed in meantime to keep real DNS server silent

DNS Cache Poisoning Method 2

- **This attack exploits a bug in some implementations of BIND**
  - A server stores in the cache anything that is contained in a DNS reply
  - If an malicious DNS server returns additional answers to a simple question the cache can be poisened
    - Some implementations will even accept answer records in DNS requests, caching the information
- **Attacker needs control over a DNS Server**
  - Additionally to an answer to a query a second entry is sent to the originator of the query
- **Stored in the DNS cache of the attacked server**
  - For the TTL of the DNS RR entry

DNS Cache Poisoning Method 3 1



Attacker

1. DNS Query
www.attack.com = ?

3. Store DNS Query-ID

2. DNS Query
www.attack.com = ?

User Trust_All

DNS-Server
dns.trustall.com

DNS-Server
dns.attack.com

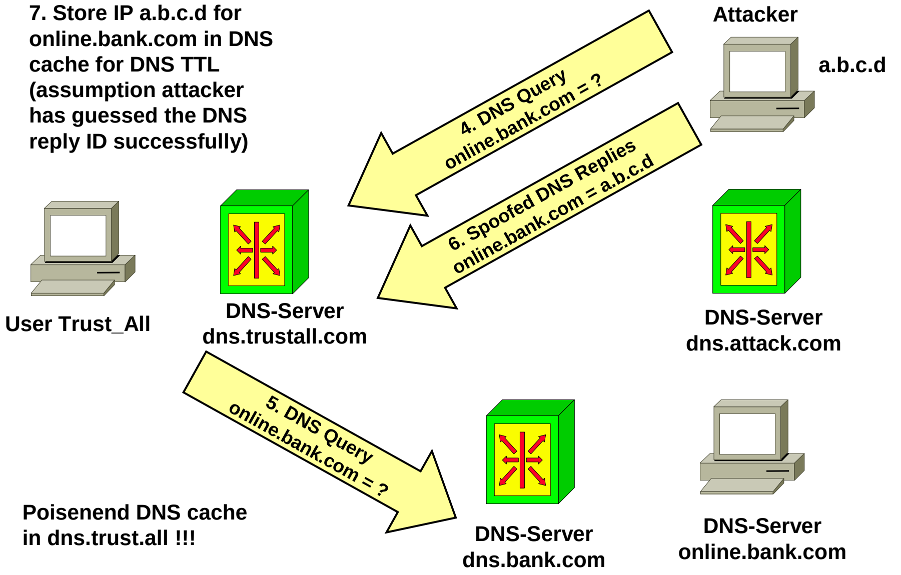
Attacker get dns.trustall.com to resolve some symbol of the attack.com domain to find out the behavior of DNS query-IDs

dns.attack.com is under attacker's full control

DNS-Server
dns.bank.com

DNS-Server
online.bank.com

DNS Cache Poisoning Method 3 2



7. Store IP a.b.c.d for online.bank.com in DNS cache for DNS TTL (assumption attacker has guessed the DNS reply ID successfully)

Attacker

a.b.c.d

4. DNS Query
online.bank.com = ?

6. Spoofed DNS Replies
online.bank.com = a.b.c.d

User Trust_All

DNS-Server
dns.trustall.com

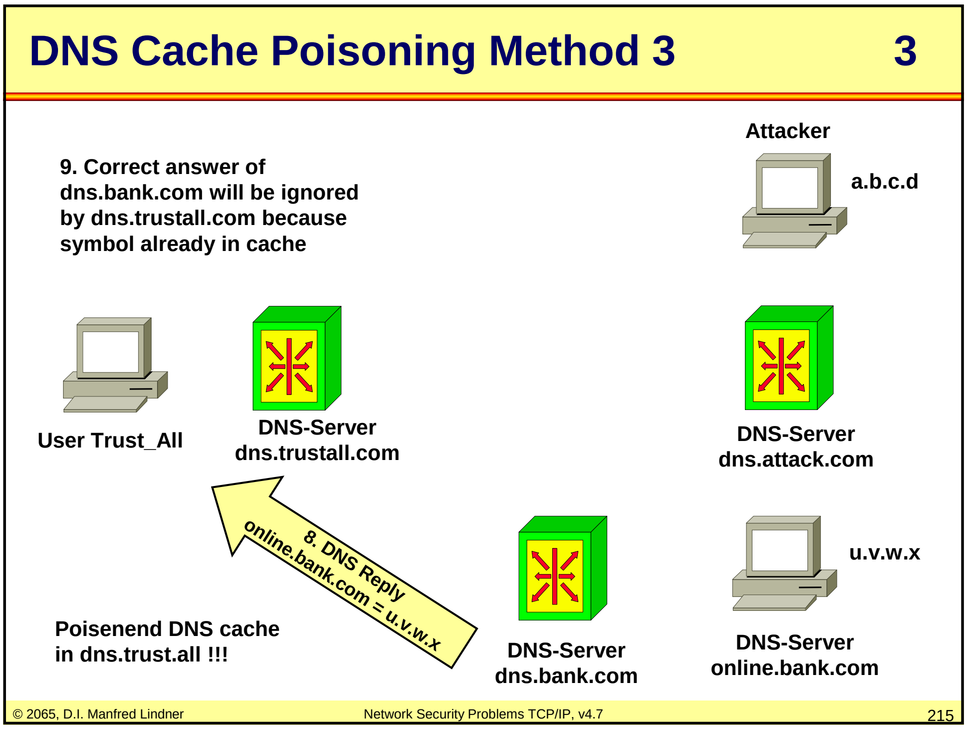
DNS-Server
dns.attack.com

5. DNS Query
online.bank.com = ?

Poisenend DNS cache in dns.trust.all !!!

DNS-Server
dns.bank.com

DNS-Server
online.bank.com



DNS Cache Poisoning Method 3

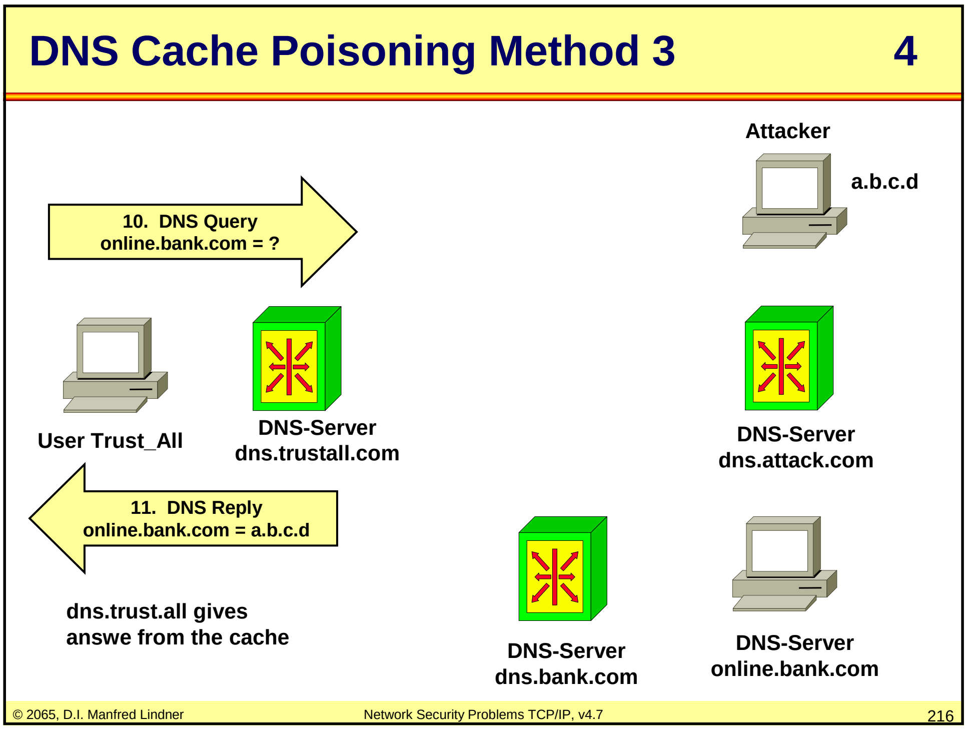
3

9. Correct answer of dns.bank.com will be ignored by dns.trustall.com because symbol already in cache

Poisenend DNS cache in dns.trust.all !!!

Attacker
a.b.c.d
User Trust_All
DNS-Server
dns.trustall.com
DNS-Server
dns.attack.com
8. DNS Reply
online.bank.com = u.v.w.x
DNS-Server
dns.bank.com
u.v.w.x
DNS-Server
online.bank.com

© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 215



DNS Cache Poisoning Method 3

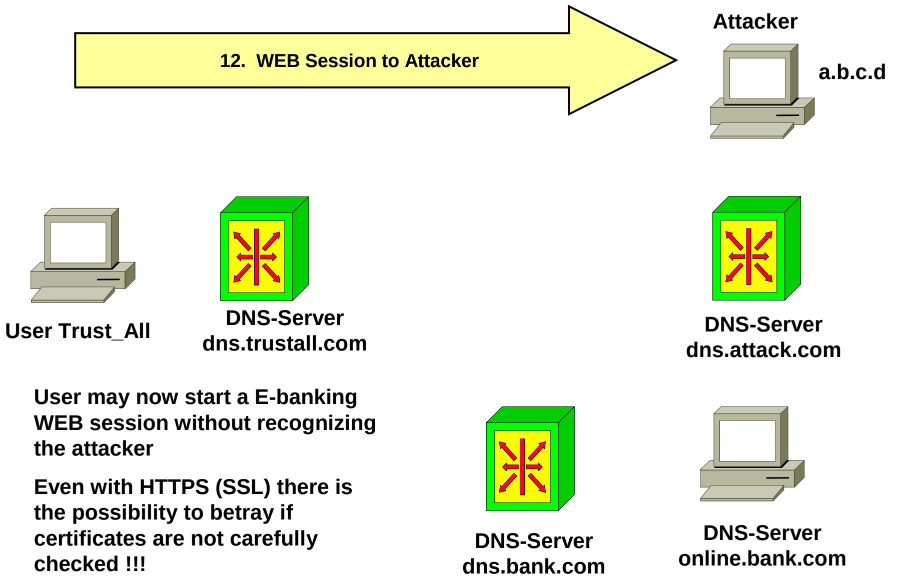
4

dns.trust.all gives answe from the cache

Attacker
a.b.c.d
10. DNS Query
online.bank.com = ?
User Trust_All
DNS-Server
dns.trustall.com
DNS-Server
dns.attack.com
11. DNS Reply
online.bank.com = a.b.c.d
DNS-Server
dns.bank.com
DNS-Server
online.bank.com

© 2065, D.I. Manfred Lindner Network Security Problems TCP/IP, v4.7 216

DNS Cache Poisoning Method 3 5


12. WEB Session to Attacker
Attacker
a.b.c.d
User Trust_All
DNS-Server
dns.trustall.com
DNS-Server
dns.attack.com
DNS-Server
dns.bank.com
DNS-Server
online.bank.com

User may now start a E-banking WEB session without recognizing the attacker

Even with HTTPS (SSL) there is the possibility to betray if certificates are not carefully checked !!!

Mitigation of DNS Cache Poisoning

- **Make prediction of DNS query-IDs harder**
  - Random number generation instead of linear sequencing
  - Upgrade BIND to version 8.x or above
- **Authentication and integrity checking for DNS queries and replies**
  - Digitally signed DNS
  - DNSSec

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

FTP-Principles 1

- **FTP uses client-server communication principle**
  - FTP (File Transfer Protocol) defined in RFC 959
- **client-server communication maintains 2 TCP connections**
  - control signals use the well known port 21
  - data stream is connected to the well known port 20 of the server (except passive mode is requested)
- **using TCP means: FTP needs no additional error recovery mechanisms to protect the data**
- **file access protection is done via login-procedure**
  - login name
  - password

FTP-Principles 2

- **after connection establishment of the control connection the client protocol interpreter (PI) and the server PI communicate on the control channel using the NVT format**
- **PI is responsible for**
  - translating the local syntax into the NVT syntax
  - issuing an appropriate action in the underlying OS (e.g. DOS command DIR -> UNIX command LS)
- **control connection provides commands from the client to the server and acknowledgements in the other direction**

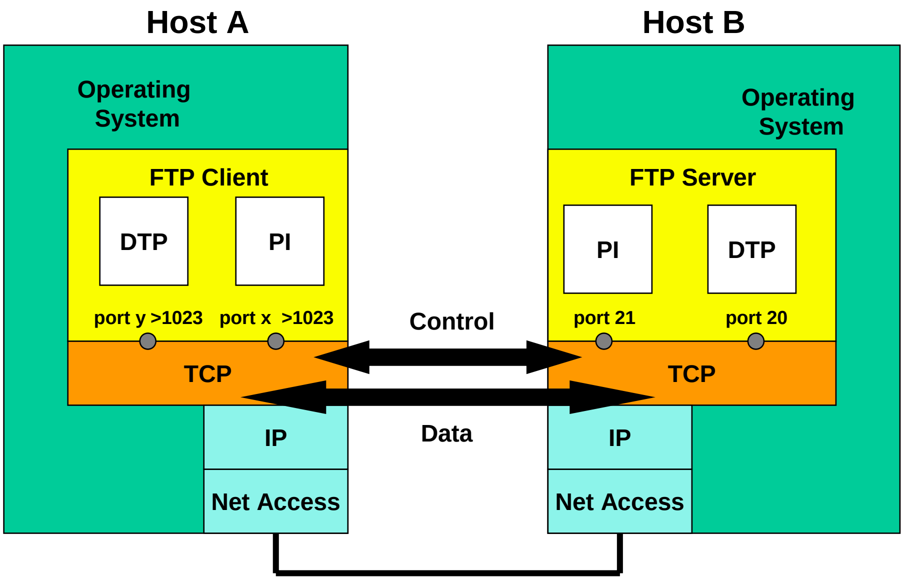
FTP-Principles 3

- **if a command issues a data transfer**
  - a client DTP (Data Transfer Process) and a server DTP are started to maintain a separate TCP- connection
- **the separate TCP connection for date transfer can be established in two ways**
  - the client specifies via control connection a portnummer to which the server setups a TCP connection from port 20 (active mode, default mode)
  - the client requests via control connection passive mode and receives a new port number (> 1023) from the server to which the client establishes the separate TCP connection (passive mode; firewall-friendly)

FTP-Principles 4

- **all data transmission flows over this channel**
- **at the end this connection is closed and the DTP's terminate**
- **this procedure is repeated for each data transmission**
  - half duplex !

FTP Internal Processes


Host A
Host B
Operating System
FTP Client
DTP
PI
port y >1023
port x >1023
TCP
IP
Net Access
Operating System
FTP Server
PI
DTP
port 21
port 20
TCP
IP
Net Access
Control
Data

Control Commands 1

- **commands of the control connection from the client to the server (NVT-format):**

Login Procedure:

– USER	provides username for login
– PASS	provides password of the user; NOTE: transmitted in plain text !!!

Directory Navigation/Creation:

– LIST	list the directory content
– CWD	change the directory
– CDUP	change to the upper directory level
– MKD	create directory
– RMD	remove directory

Control Commands 2

FTP Service :

– RETR	load file
– STOR	send file
– DELE	delete file
– RNFR	rename from (changing filenames)
– RNTO	rename to (changing filenames)
– DECE	deletes files on the server
– APPE	append to data to a file
– ALLO	allocate memory for files on the server
– NOOP	no operation; issues OK message from server
– ABOR	signals server to abort previous commands

Control Commands 3

– REIN	re-initialization; client DTP is terminated, connection to the server is still remaining
– QUIT	Logout

Transfer Parameter:

– MODE	determine transmission mode
– STRU	determine file structure
– STAT	show the connection state
– TYPE	specification of a specific data format (binary, text ASCII/EBCDIC)
– PORT	tell the socket for the data connection (forked server: only the initial announcement connection uses the well known port 20)
– PASV ….	request passive mode

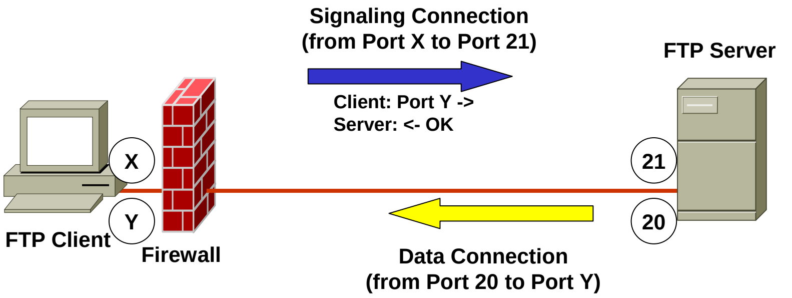
Control Commands 4

- **all commands contain the necessary arguments**
  - username, password
  - socket-ID, port-id
  - filename, directory
  - datatype:
    - ASCII, EBCDIC, Image
  - file structure:
    - file or record
  - transmission mode:
    - stream, block or compressed
- **and are completed with CR and LF**

Acknowledge Messages

- **acknowledge types of the control connection from the server to the client (NVT-format):**
  - 220, service ready, CR, LF
  - 331, user name OK, need password, CR, LF
  - 230, user logged in, proceed, CR, LF
  - 200, command OK, CR, LF
  - 150, file status OK, opening data connection, CR, LF
  - 226, closing data connection, CR, LF
  - etc.….
- **acknowledges are printed without further processing**
  - text messages for the user
  - numbers allow easy integration in programs

Operation Mode - Classic



Signaling Connection
(from Port X to Port 21)

FTP Server

Client: Port Y ->
Server: <- OK

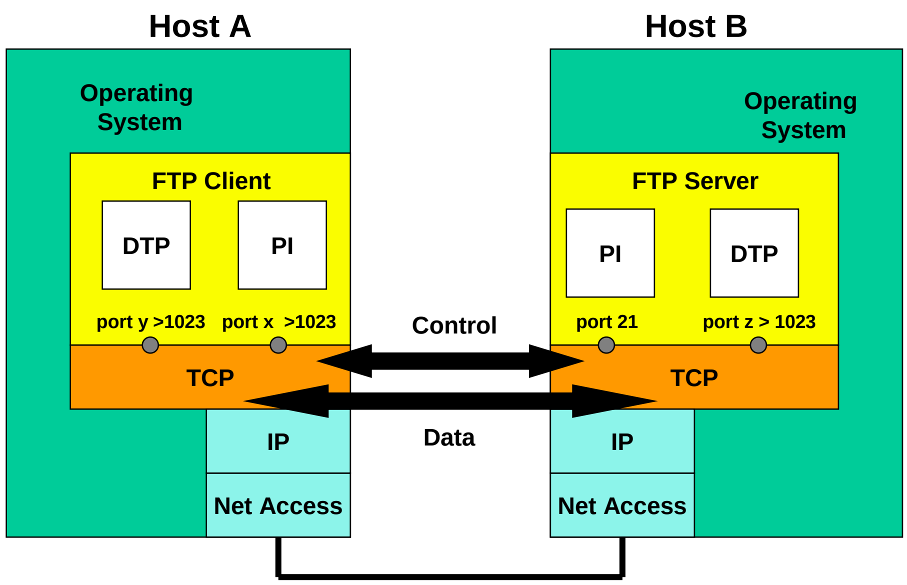
FTP Client

Firewall

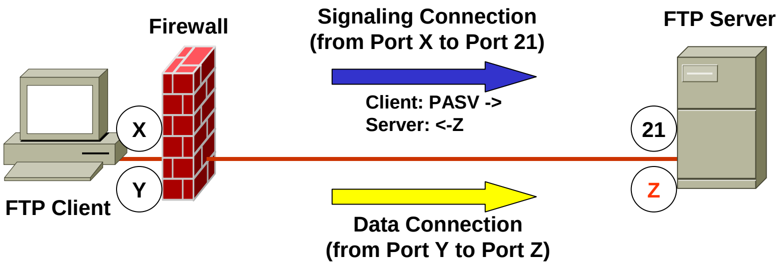
Data Connection
(from Port 20 to Port Y)

- **Firewall problems**
  - Blocks all incoming connections
- **Old Mode**

FTP Internal Processes (Passive Mode)


Host A
Operating System
FTP Client
DTP
PI
port y >1023
port x >1023
TCP
IP
Net Access
Control
Data
Host B
Operating System
FTP Server
PI
DTP
port 21
port z > 1023
TCP
IP
Net Access

Operation Mode - Passive


Firewall
FTP Client
X
Y
Signaling Connection
(from Port X to Port 21)
Client: PASV ->
Server: <-Z
Data Connection
(from Port Y to Port Z)
FTP Server
21
Z

- **Only outbound connections**
  - No Firewall problems
  - see RFC 1123, 1579, 1635
- **Best mode in secure environment**

Agenda

- **IP**
  - Review IP, ICMP
  - L3 Attacks on IP
- **TCP**
  - Review TCP
  - L3/L4 Attacks on TCP
- **UDP**
  - Review UDP
  - L3/L4 Attacks on UDP
- **DNS**
  - Review DNS, Bind, Resource Records, DNS Protocol
  - L3/L7 Attack on DNS
- **FTP**
  - Review FTP
  - FTP Bounce Attack

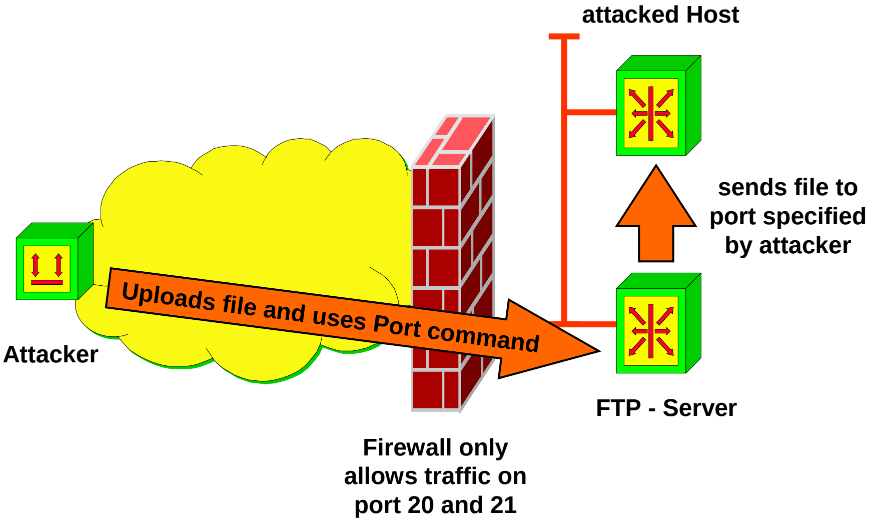
FTP Bounce Attack

- **The PORT command is used by the client**
  - to tell the server the address and port to be used when opening a data connection
  - According to the RFC 959 the address does not have to be the same as the one the client has
- **Therefore it is possible to instruct a FTP server to open a connection to a third host**

FTP Bounce Attack

- **Can be misused to perform a TCP port-scan**
  - The FTP Server appears to be the source of the scan
  - It is possible to scan „behind“ a firewall (suppose that only port 21 and 20 are open at the firewall)
- **Can be used to send data to any port**
  - If an FTP writable directory exists on the third host, a file can be transferred to a third host
  - Can be used to bypass restrictions (IP based authentication) in same way as IP spoofing

FTP Bounce Attack


attacked Host
sends file to port specified by attacker
Attacker
Uploads file and uses Port command
FTP - Server
Firewall only allows traffic on port 20 and 21