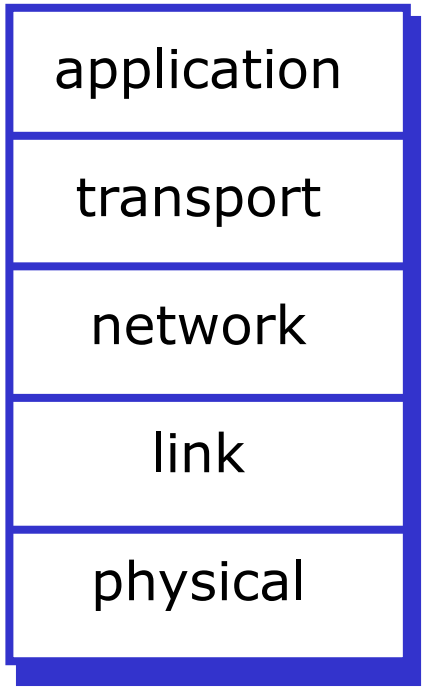
Added – TCP/IP Security

- Internet Protocol Review
- Security Problems in Internet Protocols

Internet protocol stack

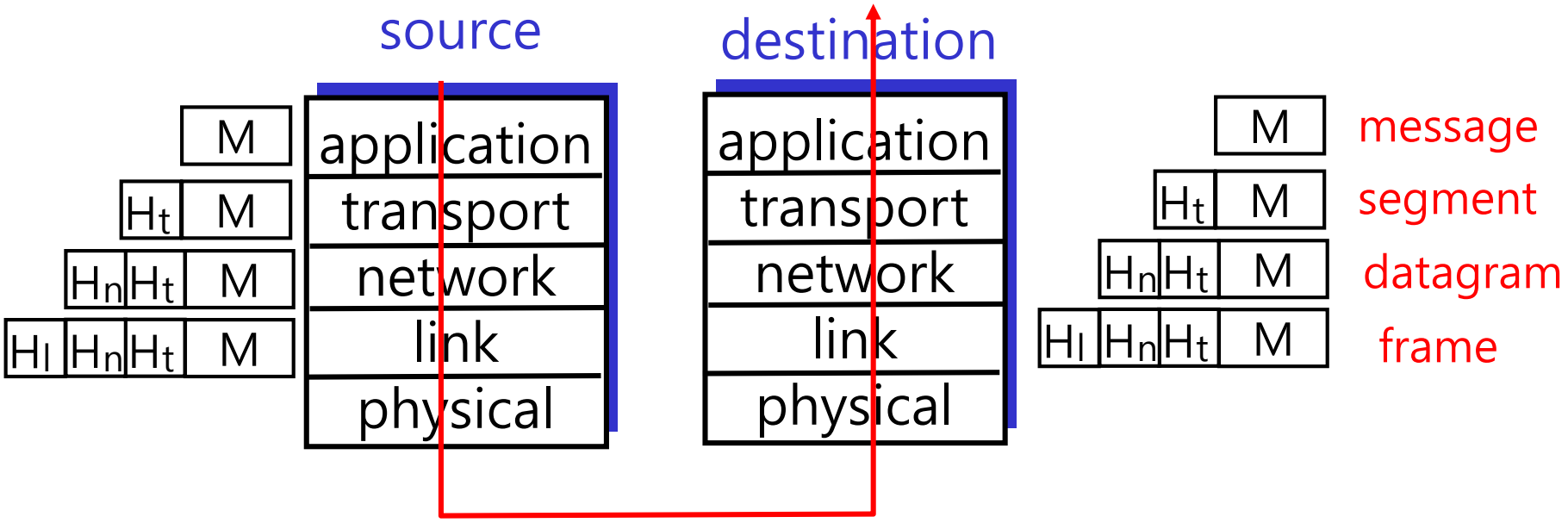
- application: supporting network applications
  - ftp, smtp, http
- transport: host-to-host data transfer services
  - tcp, udp
- network: routing of datagrams from source to destination
  - ip, routing protocols (rip, ospf, …)
- link: data transfer between neighboring network elements
  - ppp, ethernet
- physical: bits “on the wire”



Internet protocol stack

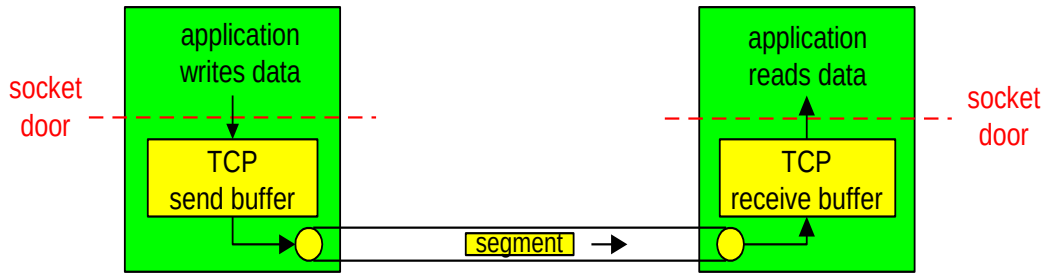
Each layer takes data from above

- adds header information to create new data unit
- passes new data unit to layer below



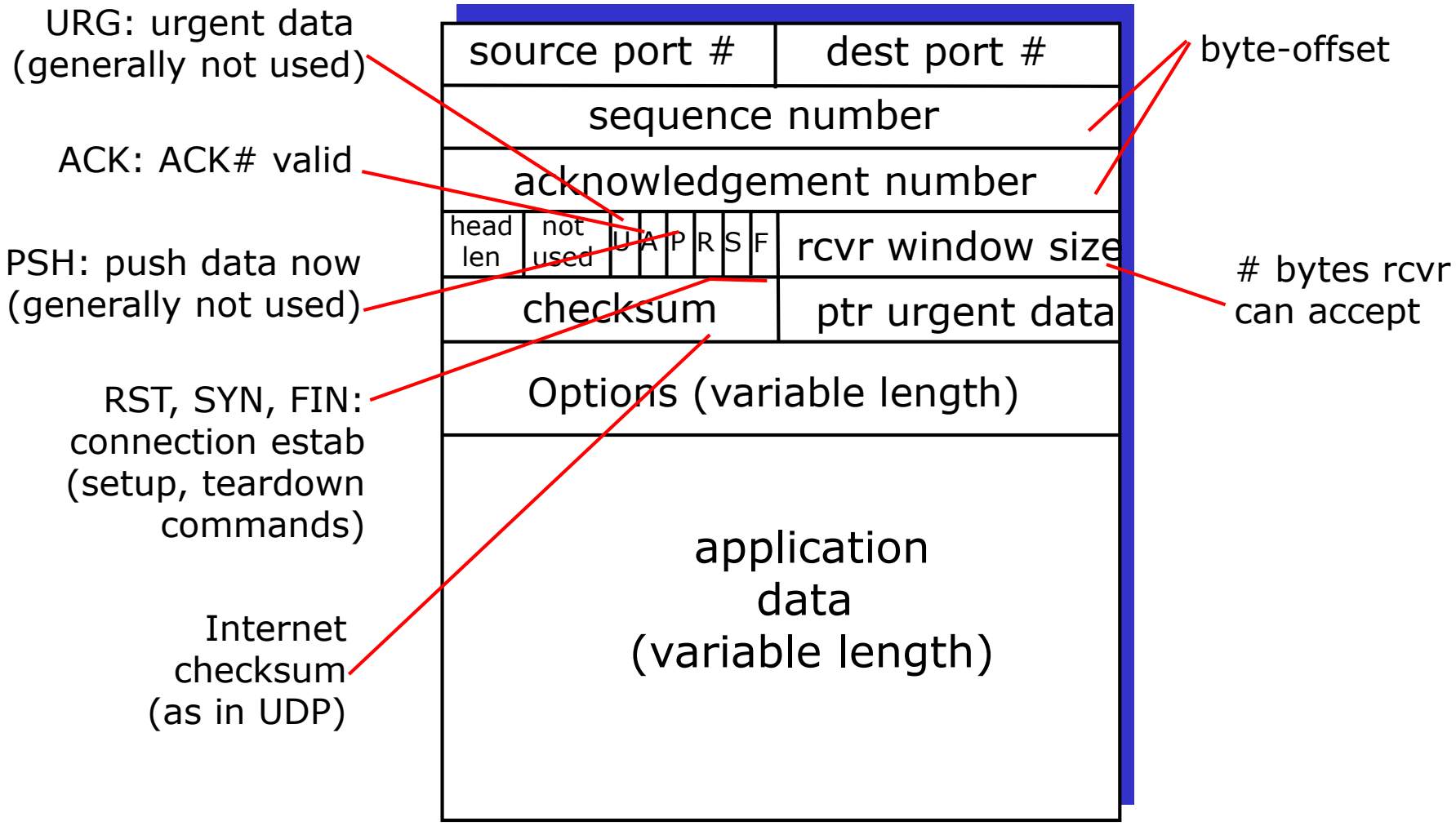
TCP Protocol Review

- point-to-point
- reliable, in-order *byte steam:*
  - no "message boundaries"
- pipelined:
  - TCP congestion and flow control set window size
- Buffering*:*
  - *send & receive buffers*



- full duplex data:
  - bi-directional data flow in same connection
- connection-oriented:
  - Handshaking: initialize sender, receiver state before data exchange
- flow controlled:
  - sender will not overwhelm receiver

TCP Protocol Review



ICMP: Internet Control Message Protocol

- used by hosts, routers, to get network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

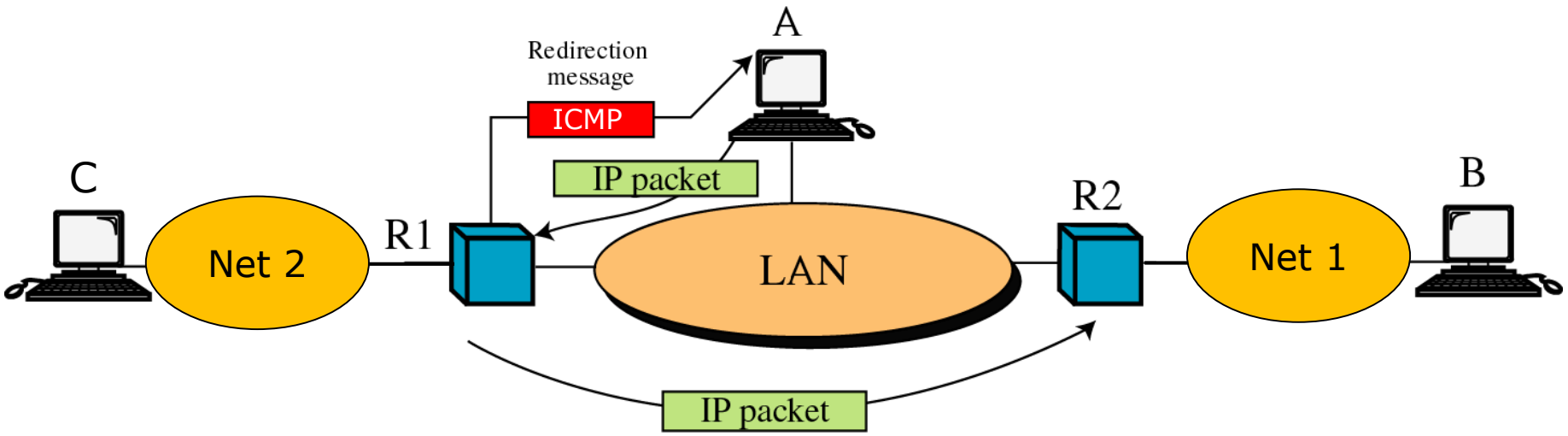
Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench(congestion control - not used)
5	0~3	icmp redirect
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

ICMP Redirects

- ICMP is used for routing error messages
  - TTL expired (traceroute)
  - Host unreachable
  - Echo request (ping program)
- Also used by default routers to redirect along quicker path

ICMP Redirect Routing

- Redirection
  - A host usually starts with a small routing table that is gradually augmented and updated
  - A ICMP redirection message is sent from a router to a host on the same local network

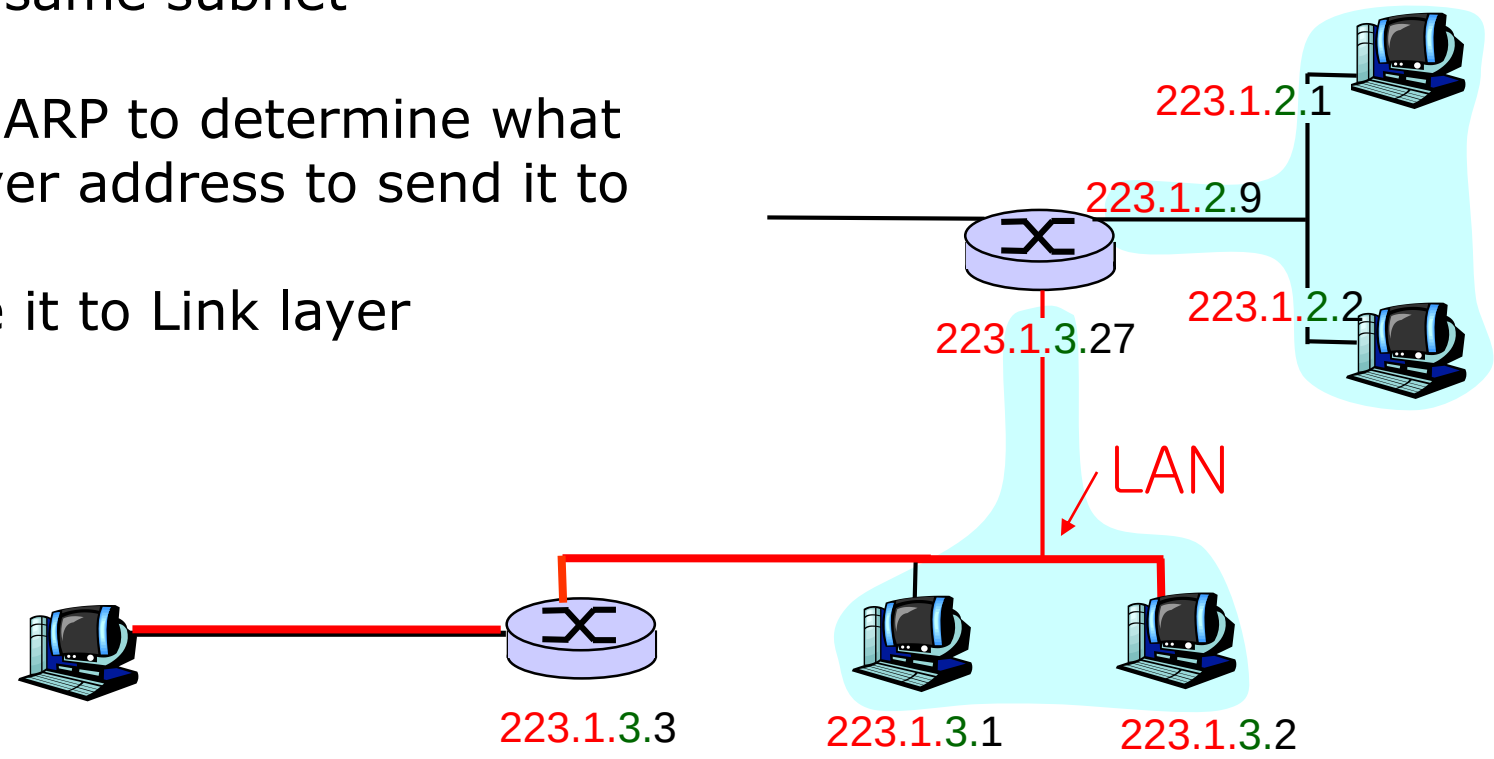


ARP: On-the-same-LAN routing

1. Route lookup determines its on the same subnet

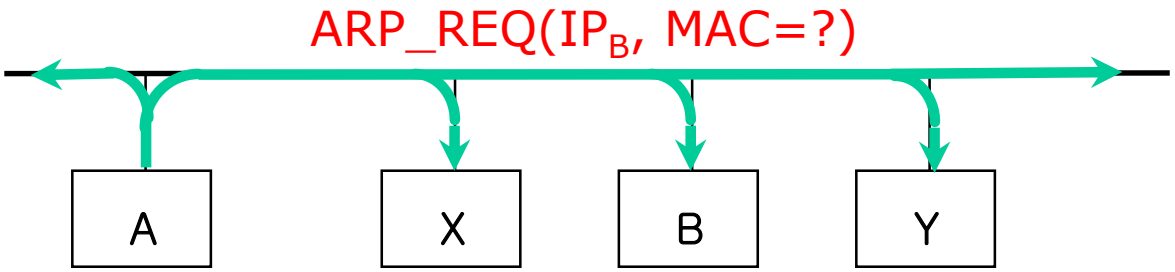
2. Use ARP to determine what link layer address to send it to

3. Give it to Link layer

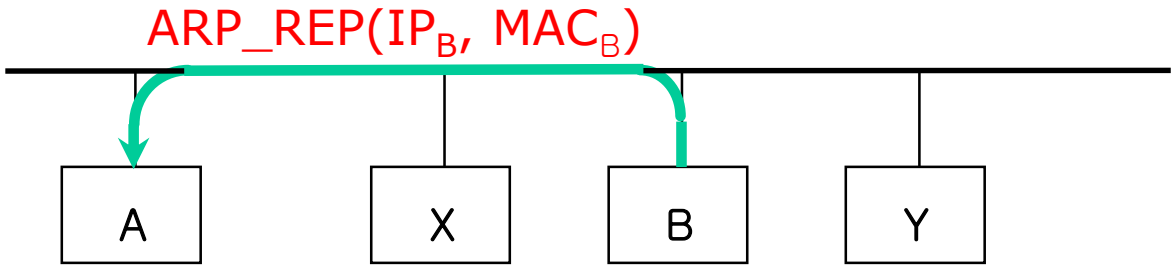


Address Resolution Protocol (ARP)

- Used in the broadcast LANs



a) host A broadcasts an ARP request containing I_B

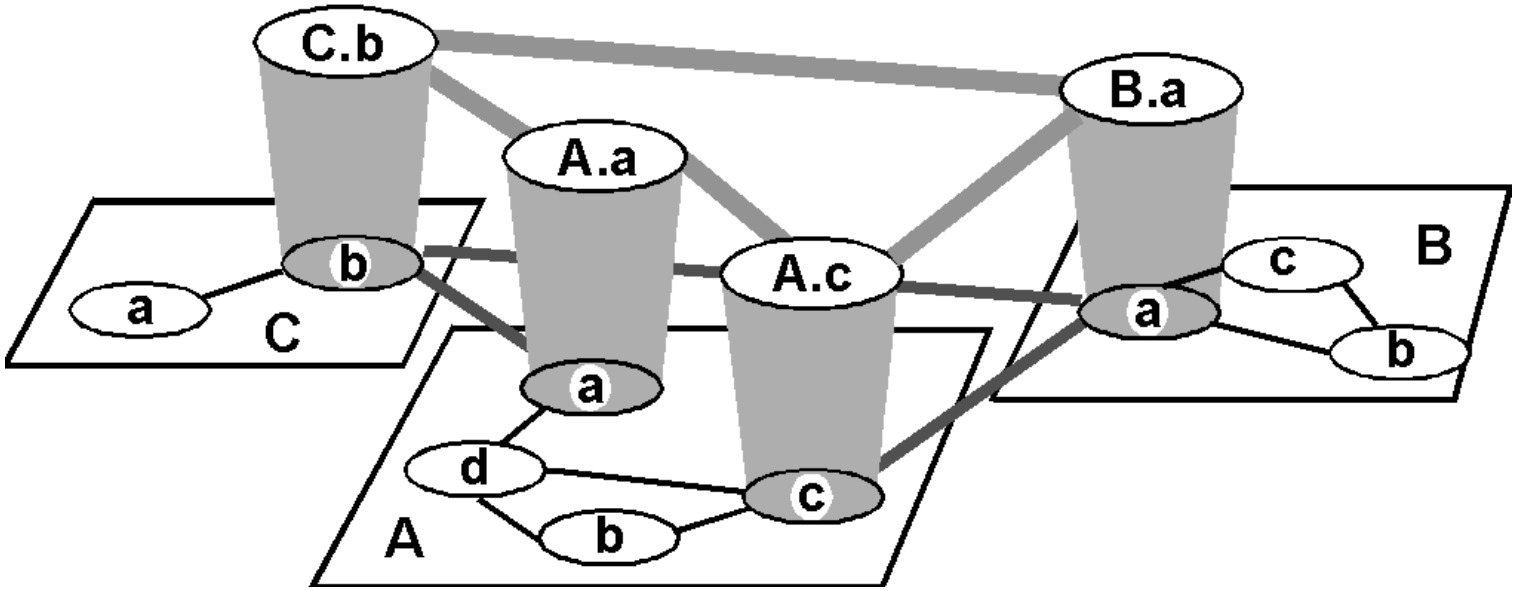


b) host B responds with an ARP reply containing the pair (I_B, P_B)

- IP address set-up using gratuitous ARP

Inter-AS routing

- AS: autonomous system
  - Keeps the routing information within its AS
  - Exports to the other AS router (exterior gateway)
- Interior routing vs. exterior routing protocols



Inter-AS routing

- BGP (Border Gateway Protocol): the de facto standard
- Path Vector algorithm: and extension of Distance Vector
- Each Border Gateway broadcast to neighbors (peers) the entire path (ie, sequence of ASs) to destination
- For example, Gateway X may store the following path to destination Z:

$$\text{Path } (X,Z) = X,Y1,Y2,Y3,\ldots,Z$$

Inter-AS routing

- Now, suppose GW X send its path to peer GW W
- GW W may or may not select the path offered by GW X, because of policy (cost, or loop prevention reasons)
- If GW W selects the path advertised by GW X, then:

Path (W,Z) = W, X,Y1,Y2,Y3,…,Z

Security Problems in Internet Protocols

- Reference: Security Problems in the TCP/IP Protocol Suite by Steve Bellovin
- R-services
- Source-routing
- ARP attacks
- Session hijacking
- TCP session stealing

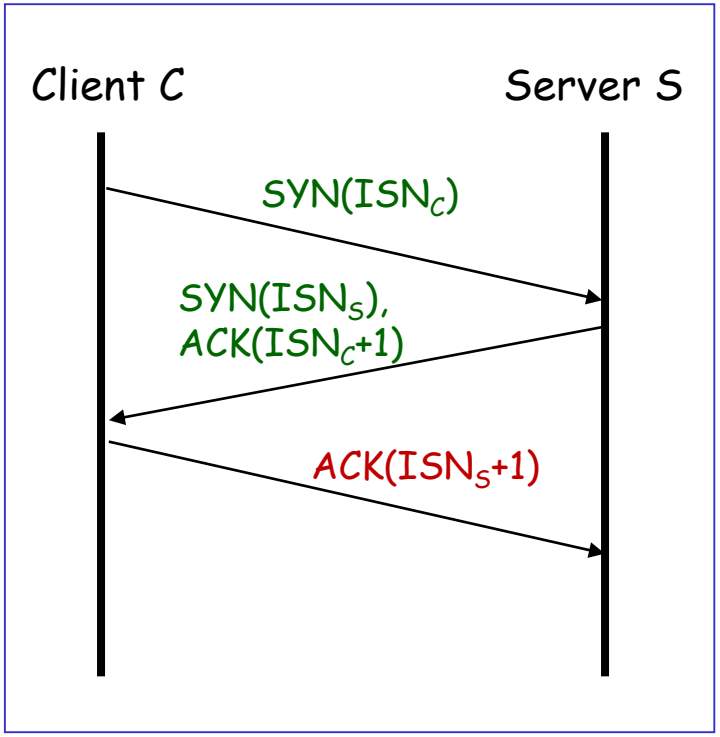
Security problems in r-services

- rsh and rcp use the *.rhosts* file in your directory, which lists hosts and accounts to allows access from without a password
- r-services: allowed by /etc/inetd
- Example *.rhosts* (or /etc/rhosts) file:

```
www.ulsan.ac.kr   mkkim
*.ulsan.ac.kr   mkkim
*   *
```

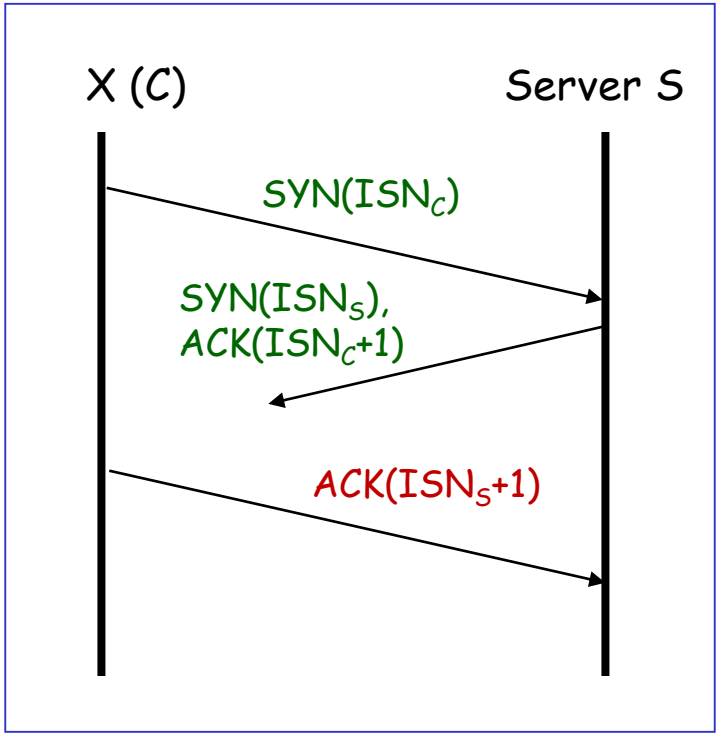
Making a Connection to rsh Server

- 3-way handshaking
  - C->S: SYN(ISN_C)
  - S->C: SYN(ISN_S), ACK(ISN_C+1)
  - C->S: ACK(ISN_S +1)
  - Client and Server exchange data (rsh command)



Making a Connection to rsh Server

- 3-way handshaking
  - X->S (spoof): SYN(ISN_C)
  - S->C: SYN(ISN_S), ACK(ISN_C+1)
  - X->S (spoof): ACK(ISN_S +1)
  - How can X know ISN_S ?



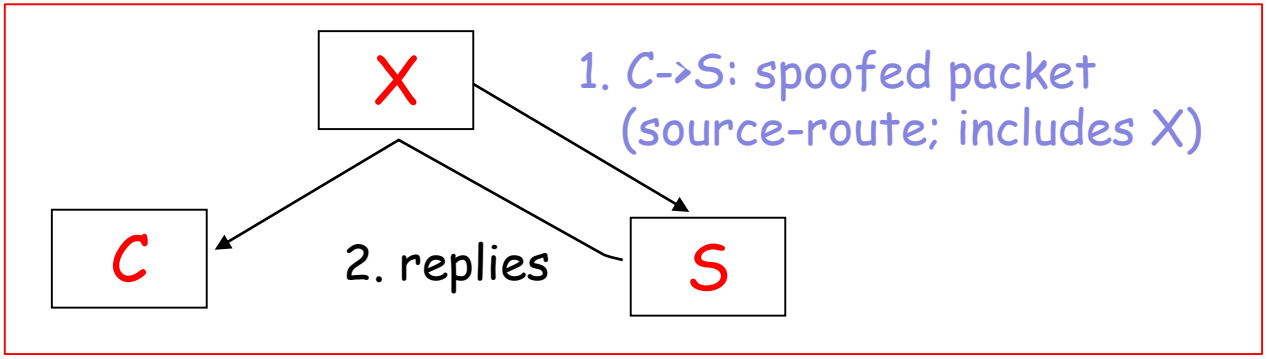
Security problems in r-services

- a machine is running rsh, how can we pretend to be another machine to gain access?

Attack	Defense
Source routing	ignore source routes
False routing table updates	secure routing protocols
Session hijacking	ssh / secure connection
ICMP redirects	ignore ICMP redirects
False ARP packets	publish ARP tables
TCP session stealing	ssh/ secure connection

Security problems in r-services

- Exploiting trusted relationships: C is a trusted host to S
- Source routing:
  - IP source-route option
  - The responder uses the source-route on the reply
  - Open a TCP connection to rshd spoofing the address of a trusted host, but include yourself in the source route

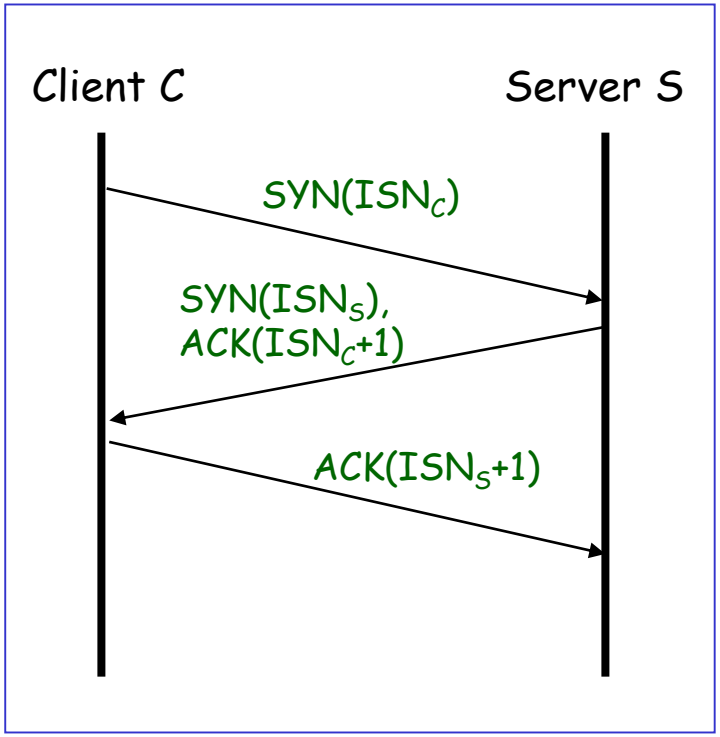


- Countermeasure: ignore source routes

Session hijacking

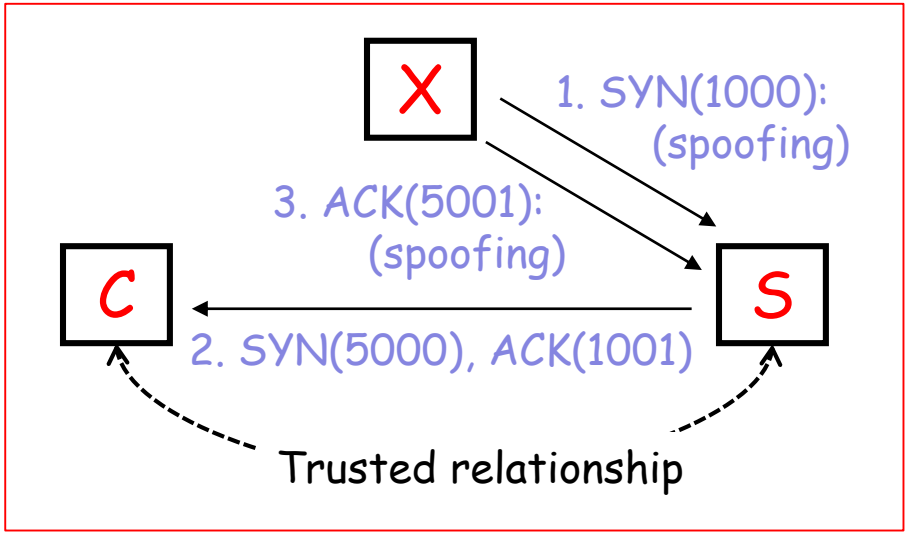
- Normal TCP operation from client, C, to server, S
  - C->S: SYN(ISN_C)
  - S->C: SYN(ISN_S), ACK(ISN_C+1)
  - C->S: ACK(ISN_S +1)
  - Client and Server exchange data

  - ISN number generation
  - 4.2BSD: increments 128/sec
    (1 for 7.8ms)
  - 4.3BSD: increments 125,000/sec
    (1 for 8us)



Session hijacking

- Attacker X knows
  - S provides R-service
  - C is a trusted host of S
- X wants to disguise as C and access to S



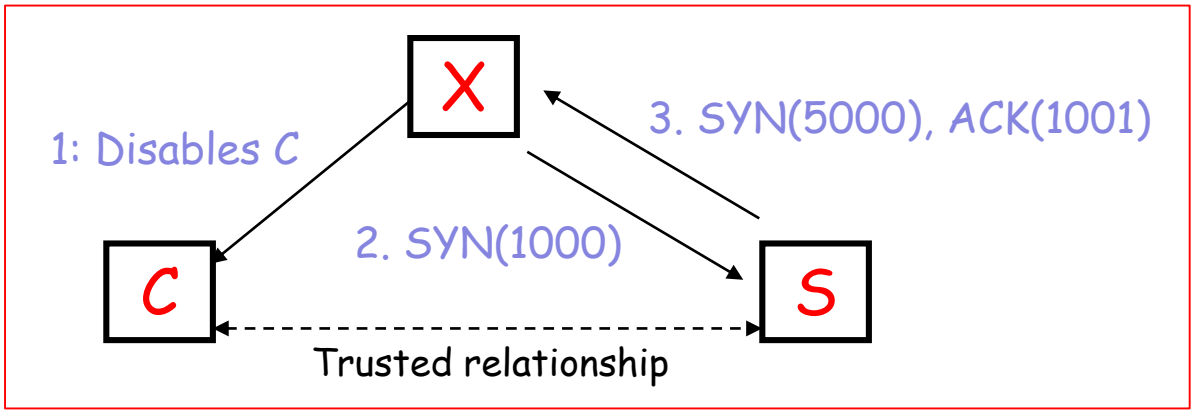
Session hijacking

Session hijacking: ISN estimation

(1) X->S: SYN(ISN_X) S: rshd server

(2) S->X: SYN(ISN_{S1}), ACK(ISN_X +1)

1. ISN estimation:



Session hijacking

- Session hijacking: session hijacking

  (3) X->S: SYN(ISN_X) [spoofs C]

  (4) S->C: SYN(ISN_{S2}), ACK(ISN_X +1)

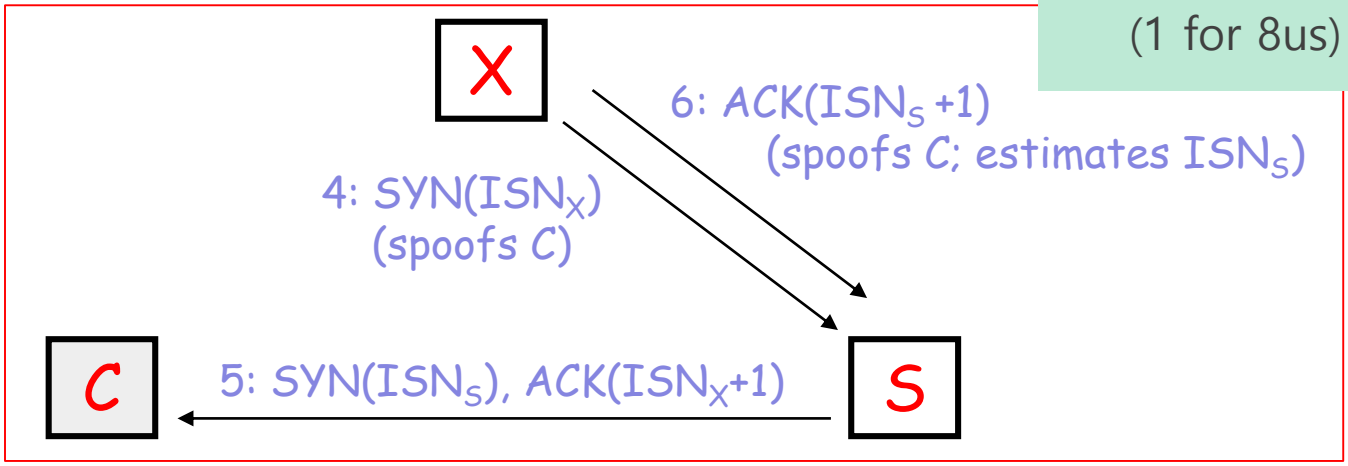
  (5) X->S: ACK(ISN_{S2} +1)

  [spoofs C; estimates ISN_{S2}]

S: rshd server

- ISN number generation
  - 4.2BSD: increments 128/sec (1 for 7.8ms)
  - 4.3BSD: increments 125,000/sec (1 for 8us)

2. Session hijacking:



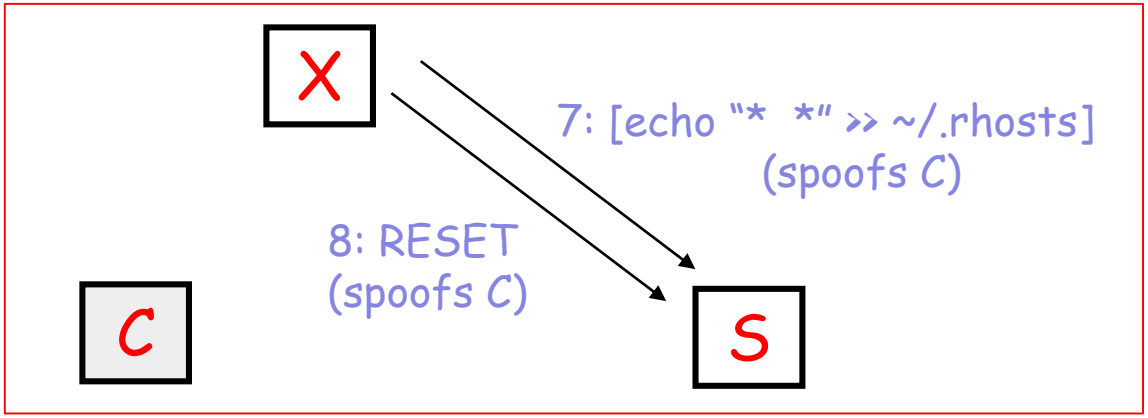
Session hijacking

- Session hijacking: execute remote commands

(6) X->S: [echo "* *" >> ~/.rhosts] [spoofs C]

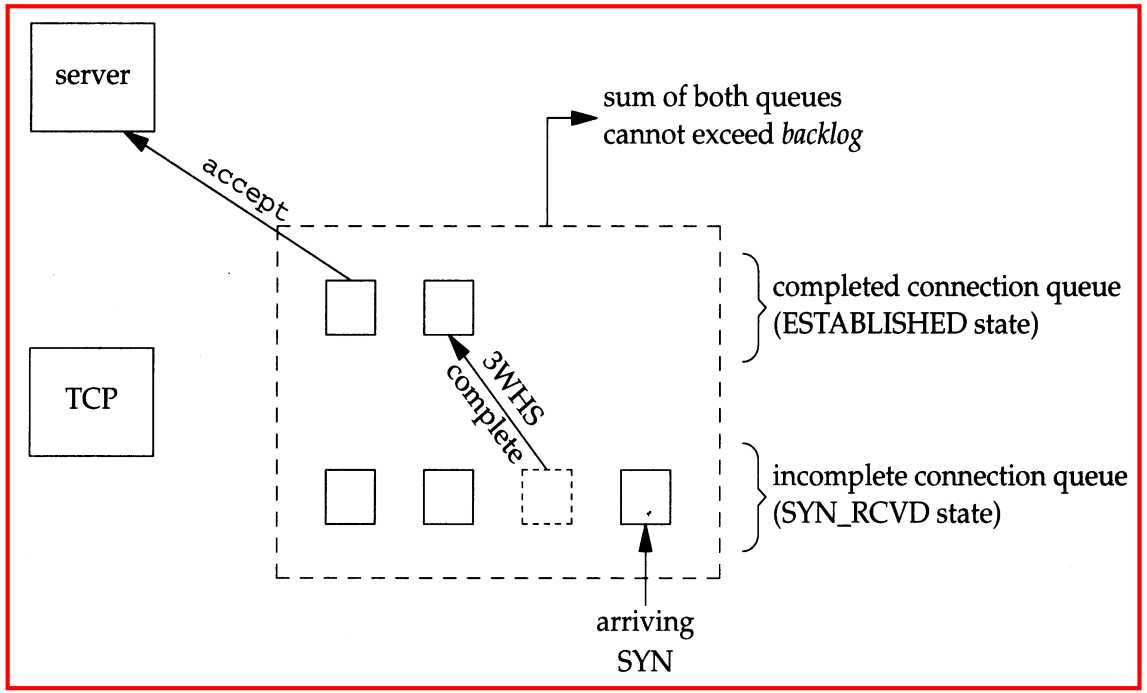
(7) X->S: RESET [spoofs C]

3. Executes remote commands: X can rlogin from anywhere in the world



Disabling hosts: SYN Flooding DoS

- Send lots of spoofed SYN packets to a victim host
- TCP connection queue
  - Kernel maintains two queues for each listening socket:
  - Incomplete conn. queue: an entry for each conn. in SYN_RCVD state
  - Completed conn. queue: an entry for each conn. in ESTABLISHED state
  - accept returns first entry on the completed queue



Attacking Routing to Exploit rsh

- Dynamic routing updates
  - OSPF: link-state algorithm
  - RIP: distance vector algorithm
- Attacker injects a fake RIP update msg. stating it has a good path to host C
  - All subsequent packets to C will be routed to the attacker
  - The attacker initiates connection to rshd of the server (spoofing C)
- Defense: uses secure routing protocols
  - Only accept authenticated updates
  - Requires key management

ICMP Attack

- ICMP redirect: forces a machine to route thru you
  - Requires an existing connection
  - Open a spoofed connection to the host you want to attack
  - Then send a spoofed ICMP redirect to the victim redirecting it to the gateway you've compromised
- Others
  - ICMP destination unreachable
  - Frequent ICMP source quenches

ARP Attacks

- When a machine sends an ARP request out, you could answer that you own the address
- Unfortunately, ARP just accepts replies without requests!
  - Just send a spoofed reply message saying your MAC address owns a certain IP address
  - Repeat frequently so that cache doesn't timeout
- Messages are routed through you to sniff or modify

ARP Spoofing - Countermeasures

- "Publish" MAC address of router/default gateway and trusted hosts to prevent ARP spoofing
  - Statically defining the IP to MAC address mapping
  - (e.g.) arp -s 203.250.77.254 00:01:02:03:04:ab pub

```
C:\Windows\system32>arp -a

Interface: 192.168.43.160 --- 0x15
  Internet Address      Physical Address      Type
  192.168.43.1          70-b7-aa-90-a3-07     dynamic
  192.168.43.160        00-aa-00-62-c6-09     static
  192.168.43.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

TCP Session Stealing

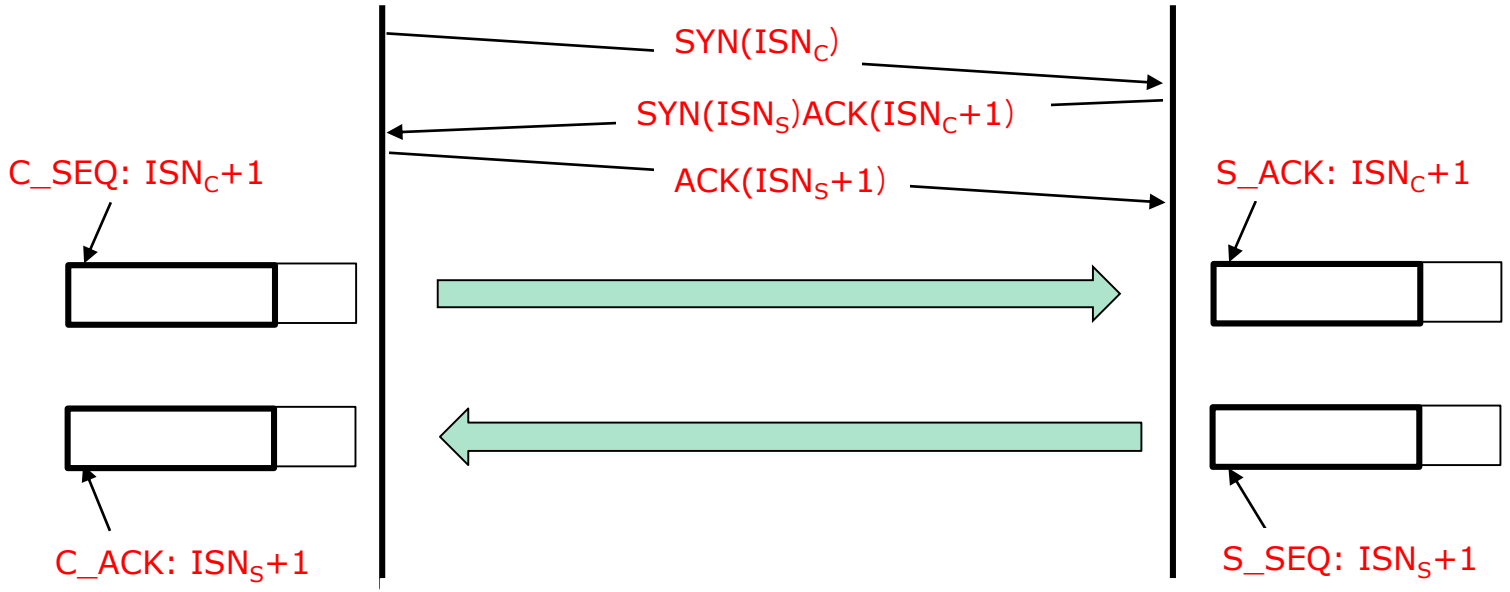
- Reference: "A Simple Active Attack Against TCP" by Laurent Joncheray, *USENIX Symposium,* June 1995

- Active attack using TCP desynchronized states
  - attacker is in the path b/w the client and server (attacker can sniff all the packets and inject some spoofed packets)
  - Steps:

    1. attacker sniffs the communication b/w the two
    2. attacker disables the communication by desynchronizing the client and the server
    3. attacker injects spoofed packets acceptable for both ends

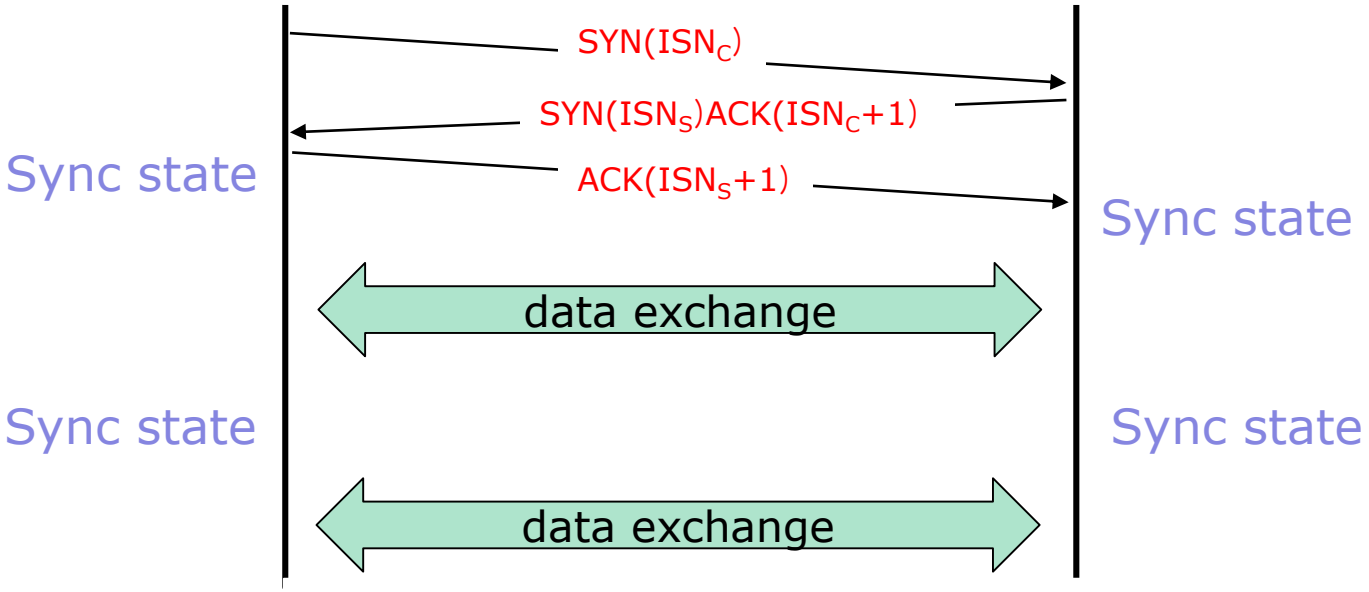
TCP Session Stealing

- Initial state after connection setup: synchronized state b/w client C and server S
  - S_SEQ = C_ACK and C_SEQ = S_ACK



TCP Session Stealing

- Desynchronized state b/w client C and server S
  - Both in “Established state”
  - No data is being sent (stable state)
  - S_SEQ ≠ C_ACK and C_SEQ ≠ S_ACK



TCP Session Stealing

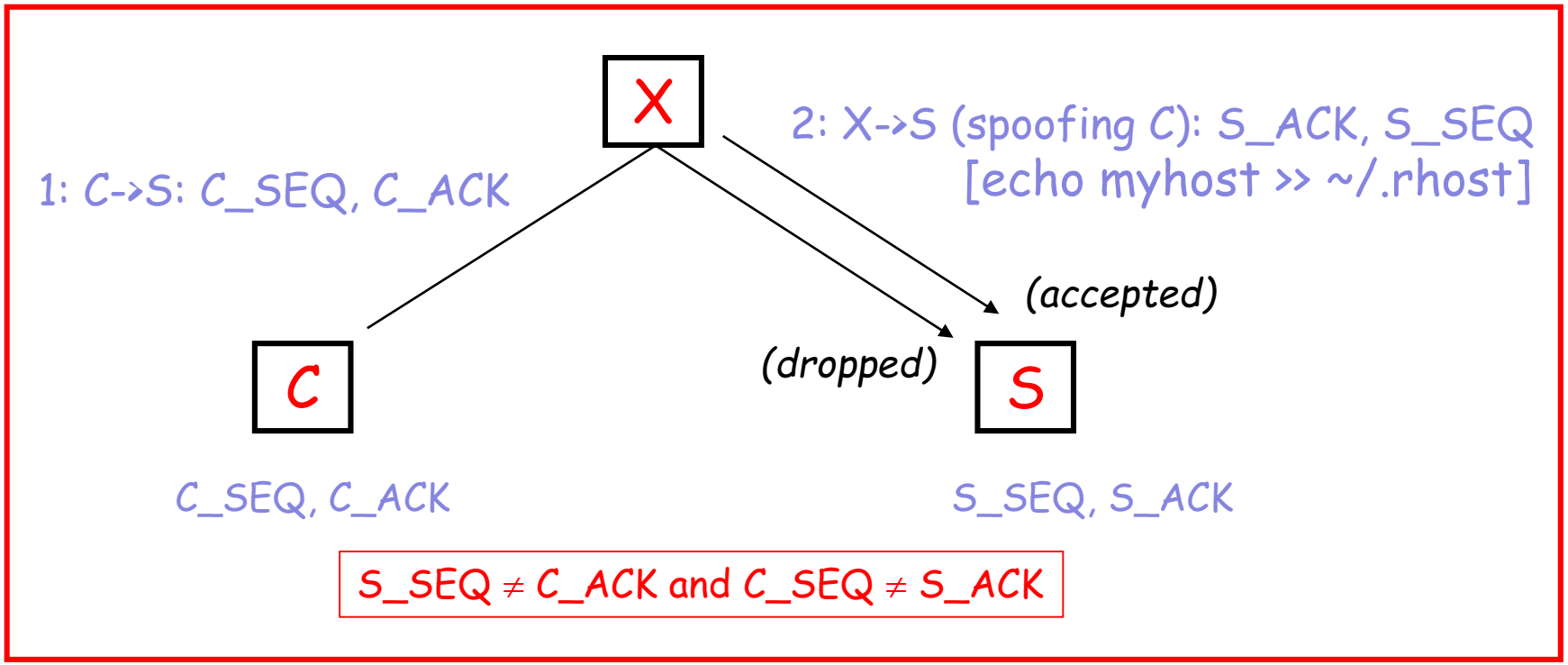
- In Desynchronized state b/w client C and server S

C→S connection: C_SEQ ≠ S_ACK

- When (C_SEQ > S_ACK + W) or (C_SEQ < S_ACK) :
  - packet is dropped
- When S_ACK < C_SEQ < S_ACK + W:
  - packet is accepted (buffered) but not sent to process
- In both cases, ACK(S_ACK) is sent : ACK packet with (S_SEQ, S_ACK)

TCP Session Stealing

- the attacker knows the desynchronized state, and can send any acceptable data to the server
  - E.g. [echo myhost >> ~/.rhost] for rlogin



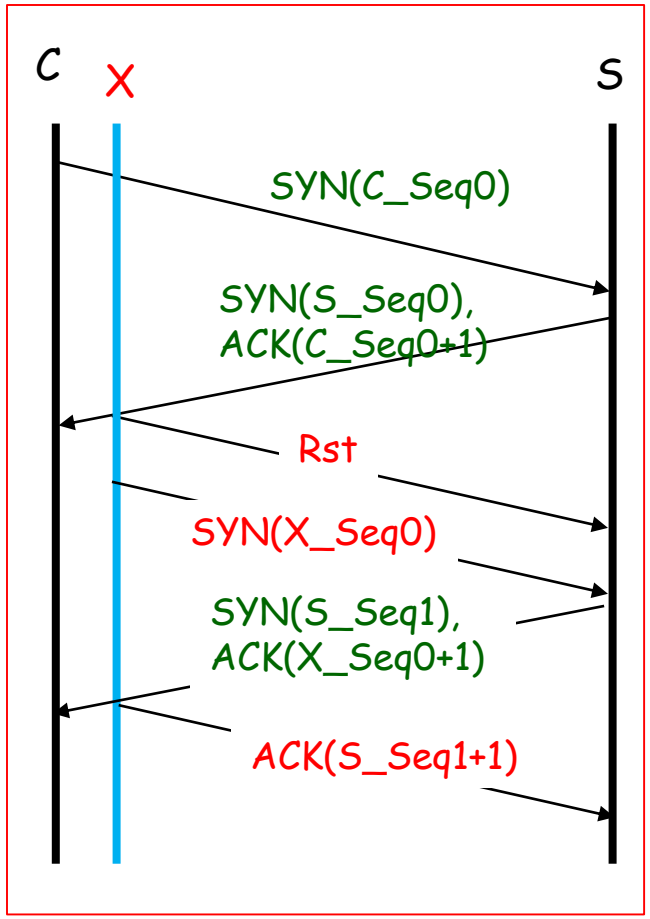
Desynchronization

Early desynchronization

1. C->S(Syn): C_Seq0 ; C: Syn_Sent
2. S->C(Syn/Ack): S_Seq0, C_Seq0+1
   ; S: Syn_Rcvd
   ; C: Established (C_Seq0+1, S_Seq0+1)

(before the packet *C->S(Ack): S_Seq0+1*)

3. X->S(spoofing C, Rst)
4. X->S(spoofing C, Syn): X_Seq0
   ; the same port # used in (1)
5. S->C(Syn/Ack): S_Seq1, X_Seq0+1
6. X->S(spoofing C, Ack): S_Seq1+1
   ; S: Established (S_Seq1+1, X_Seq0+1)



The Attack

- Null data desynchronization
  1. The attacker watches the session without interfering
  2. During a quiet period, the attacker sends a large amount of null data (IAC,NOP for telnet): nothing happens, server only changes the TCP Ack number
  3. Now, when the client sends data, it is dropped by the server because it's lower than the server's window
  4. The attacker does the same with the client
- Defense: ssh connection, or IPsec