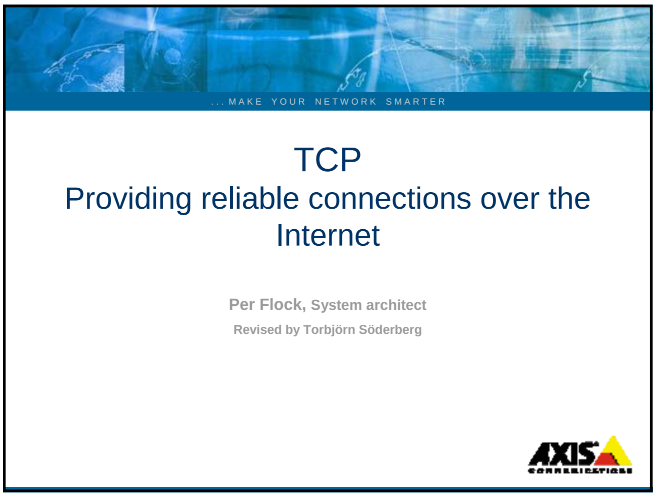
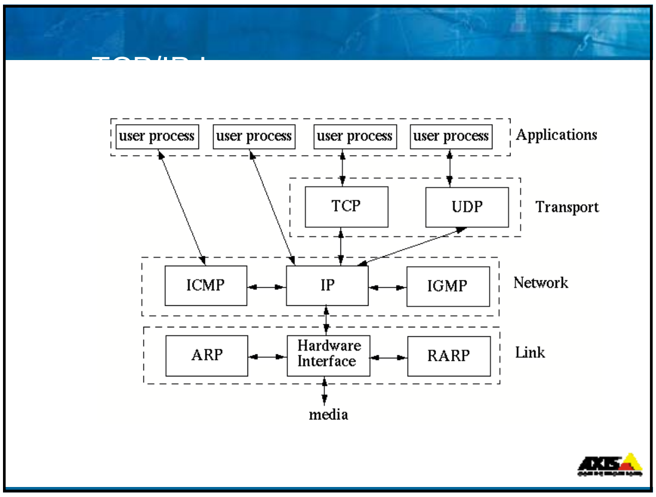


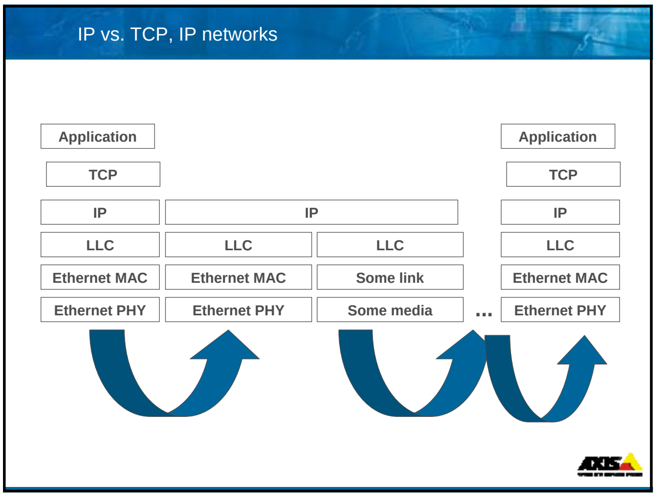
...MAKE YOUR NETWORK SMARTER

TCP
Providing reliable connections over the Internet

Per Flock, System architect

Revised by Torbjörn Söderberg





IP vs. TCP, IP networks

Application				Application
TCP				TCP
IP	IP			IP
LLC	LLC	LLC		LLC
Ethernet MAC	Ethernet MAC	Some link		Ethernet MAC
Ethernet PHY	Ethernet PHY	Some media	...	Ethernet PHY

TCP

- User Datagram Protocol (UDP)
- Transport Control Protocol (TCP)
- TCP State machine
- Reliability through acknowledgement
- Performance using windows
- Congestion avoidance
- Deadlock avoidance
- Hack attack

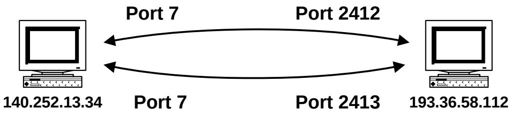
User Datagram Protocol (UDP)

- Application level end-to-end connection
- Unreliable datagram delivery
- Applications: BOOTP/DHCP, DNS, SNMP, NFS

Application level connection

Source port	Destination port
16 bit UDP length	UDP checksum
DATA	

The UDP header



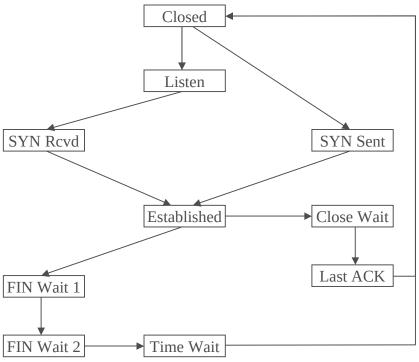
Transport Control Protocol (TCP)

- Connection oriented reliable byte stream
- TCP splits the byte stream into segments
- Every byte in the stream has a sequence number

TCP Header

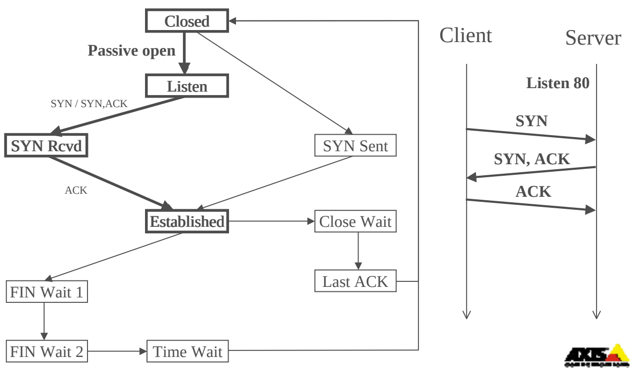
<table>
<tr><td colspan="8">0</td><td>15 16</td><td>31</td></tr>
<tr><td colspan="8">Source port</td><td colspan="2">Destination port</td></tr>
<tr><td colspan="10">Sequence number</td></tr>
<tr><td colspan="10">Acknowledgment number</td></tr>
<tr><td>Header Length (4)</td><td>Reserved (6)</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="2">Window size</td></tr>
<tr><td colspan="8">Checksum</td><td colspan="2">Urgent pointer</td></tr>
<tr><td colspan="10">Options (if any)</td></tr>
<tr><td colspan="10">Data</td></tr>
</table>

TCP State machine



- Establish connection
- Data transfer
- Terminate connection

TCP State machine



Closed

Passive open

Listen

SYN / SYN,ACK

SYN Rcvd

SYN Sent

ACK

Established

Close Wait

Last ACK

FIN Wait 1

FIN Wait 2

Time Wait

Client

Server

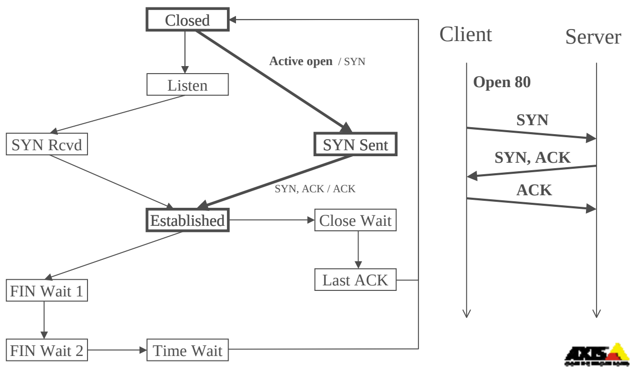
Listen 80

SYN

SYN, ACK

ACK

TCP State machine



Closed

Active open / SYN

Listen

SYN Rcvd

SYN Sent

SYN, ACK / ACK

Established

Close Wait

Last ACK

FIN Wait 1

FIN Wait 2

Time Wait

Client

Server

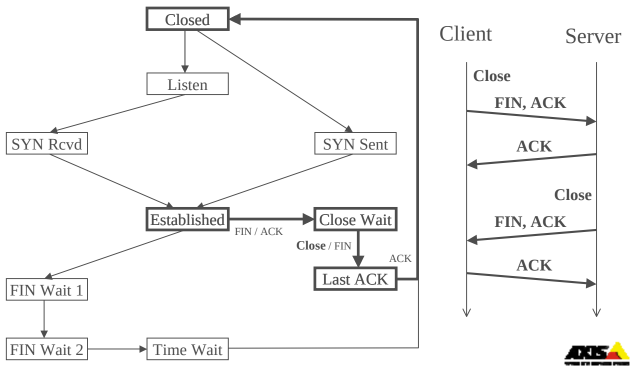
Open 80

SYN

SYN, ACK

ACK

TCP State machine



Closed

Listen

SYN Rcvd

SYN Sent

Established

FIN / ACK

Close Wait

Close / FIN

ACK

Last ACK

FIN Wait 1

FIN Wait 2

Time Wait

Client

Server

Close

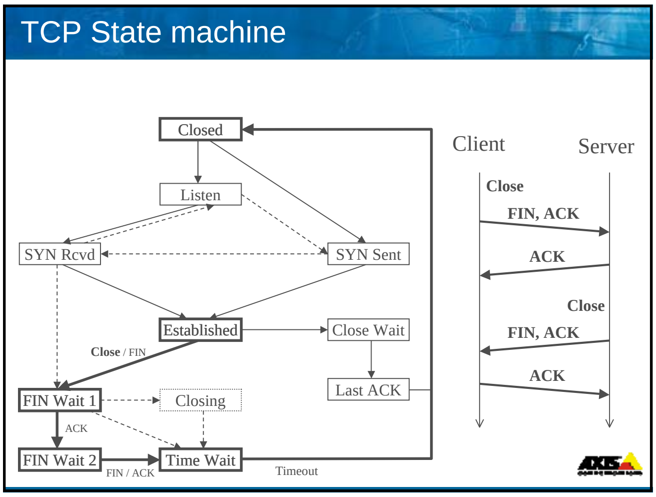
FIN, ACK

ACK

Close

FIN, ACK

ACK



TCP State machine

Closed

Listen

SYN Rcvd

SYN Sent

Established

Close Wait

Close / FIN

Last ACK

FIN Wait 1

Closing

ACK

FIN Wait 2

Time Wait

FIN / ACK

Timeout

Client

Server

Close

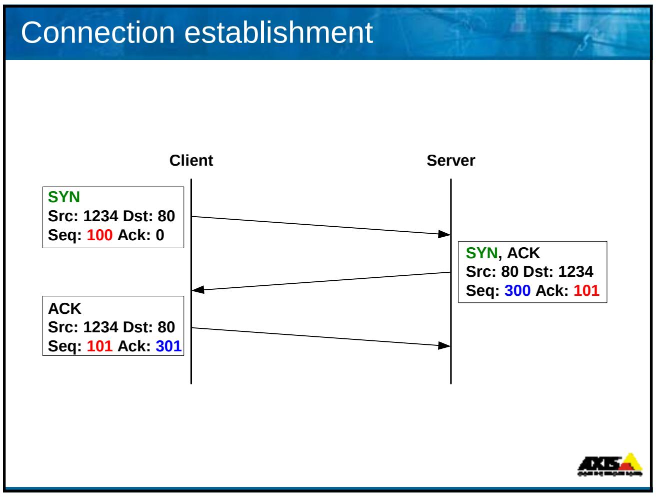
FIN, ACK

ACK

Close

FIN, ACK

ACK



Connection establishment

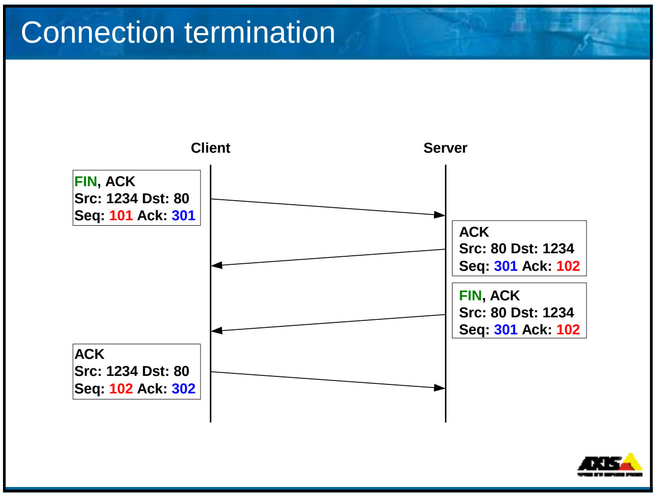
Client

Server

SYN
Src: 1234 Dst: 80
Seq: 100 Ack: 0

SYN, ACK
Src: 80 Dst: 1234
Seq: 300 Ack: 101

ACK
Src: 1234 Dst: 80
Seq: 101 Ack: 301



Connection termination

Client

Server

FIN, ACK
Src: 1234 Dst: 80
Seq: 101 Ack: 301

ACK
Src: 80 Dst: 1234
Seq: 301 Ack: 102

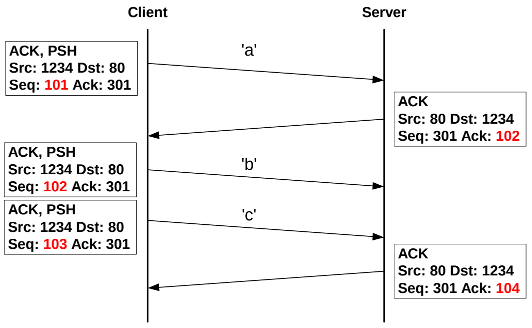
FIN, ACK
Src: 80 Dst: 1234
Seq: 301 Ack: 102

ACK
Src: 1234 Dst: 80
Seq: 102 Ack: 302

Acknowledgment

- Reliability through acknowledgement
- If sent data is not acknowledged it is retransmitted
- Acknowledgments are piggy-backed on outgoing traffic
- Delayed ACK, waits ~200ms hoping for outgoing traffic

Interactive data flow



Client

Server

ACK, PSH
Src: 1234 Dst: 80
Seq: 101 Ack: 301

'a'

ACK
Src: 80 Dst: 1234
Seq: 301 Ack: 102

ACK, PSH
Src: 1234 Dst: 80
Seq: 102 Ack: 301

'b'

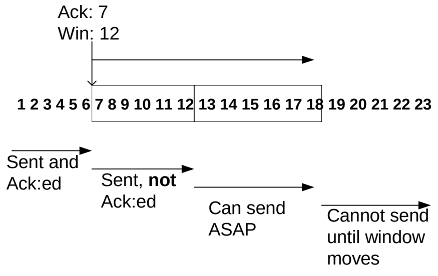
ACK, PSH
Src: 1234 Dst: 80
Seq: 103 Ack: 301

'c'

ACK
Src: 80 Dst: 1234
Seq: 301 Ack: 104

TCP bulk data flow

Sliding window



Ack: 7
Win: 12

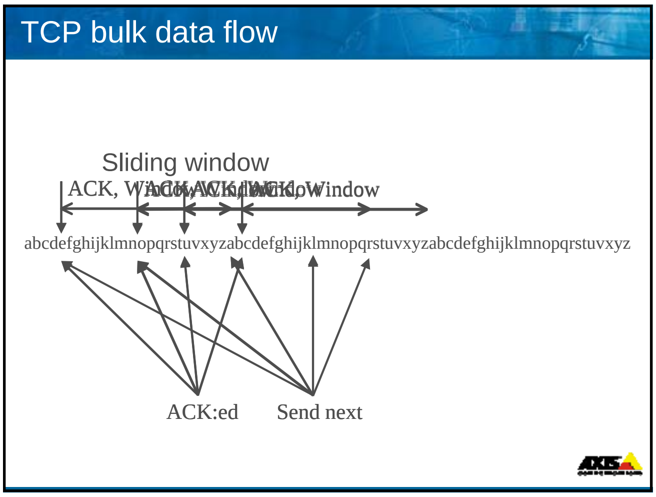
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23

Sent and Ack:ed

Sent, **not** Ack:ed

Can send ASAP

Cannot send until window moves



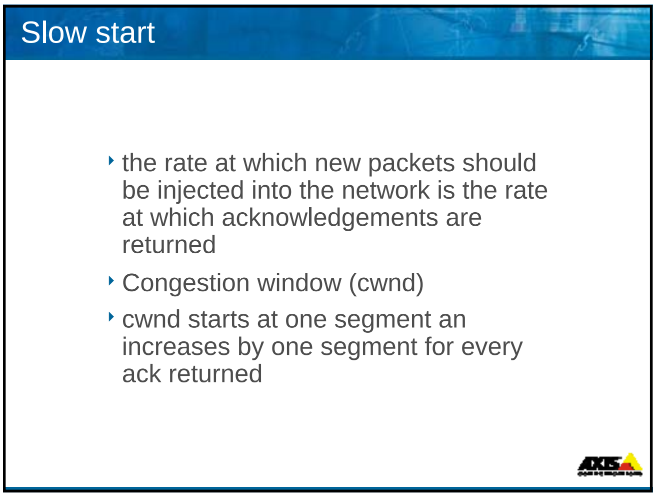
TCP bulk data flow

Sliding window

ACK, Window ACK, Window ACK, Window ACK, Window

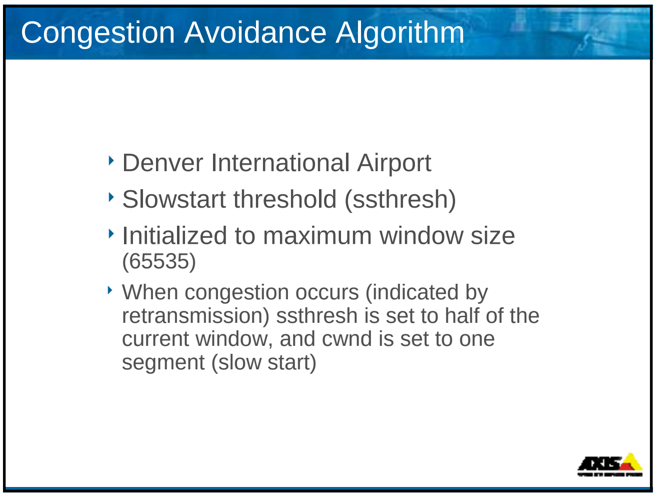
abcdefghijklmnopqrstuvxyzabcdefghijklmnopqrstuvxyzabcdefghijklmnopqrstuvxyz

ACK:ed Send next



Slow start

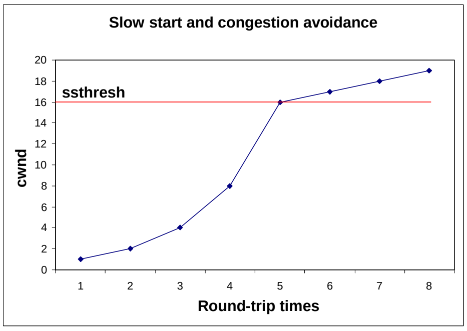
- the rate at which new packets should be injected into the network is the rate at which acknowledgements are returned
- Congestion window (cwnd)
- cwnd starts at one segment an increases by one segment for every ack returned



Congestion Avoidance Algorithm

- Denver International Airport
- Slowstart threshold (ssthresh)
- Initialized to maximum window size (65535)
- When congestion occurs (indicated by retransmission) ssthresh is set to half of the current window, and cwnd is set to one segment (slow start)

Congestion Avoidance Algorithm

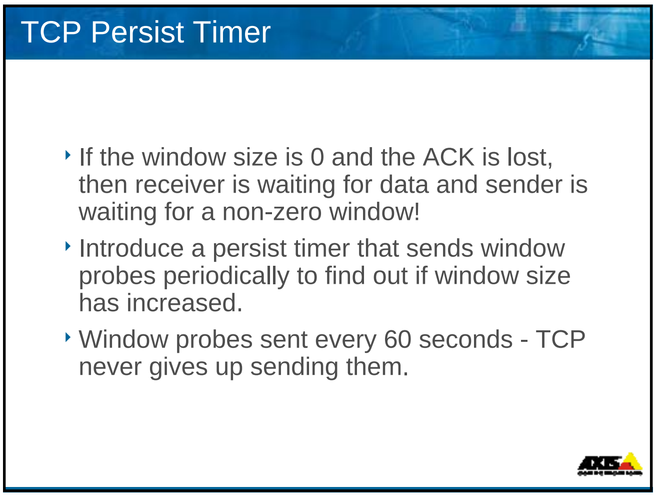


Round-Trip Time measurement

- Start timer for an outgoing segment, stop when the segment is acknowledged.
- Smoothed RTT, R
  $R = \alpha R + (1 - \alpha)M$
  - Measured time, M
  - $\alpha = 0.9$
- Retransmission timeout, RTO
  $RTO = R\beta$
  - $\beta = 2$

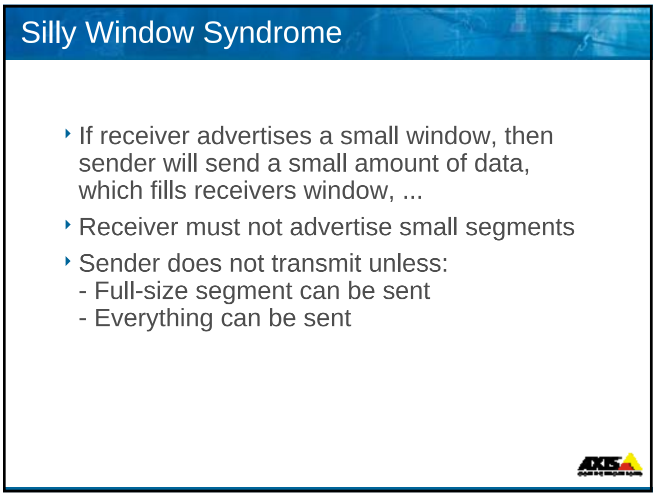
Fast retransmit and fast recovery

- Generate an immediate ACK (duplicate ACK) when an out-of-sequence segment is received
- Upon receiving 3 duplicate ACK:s the sender retransmits the lost segment without waiting for retransmission timeout
- Sender performs congestion avoidance, but not slow start



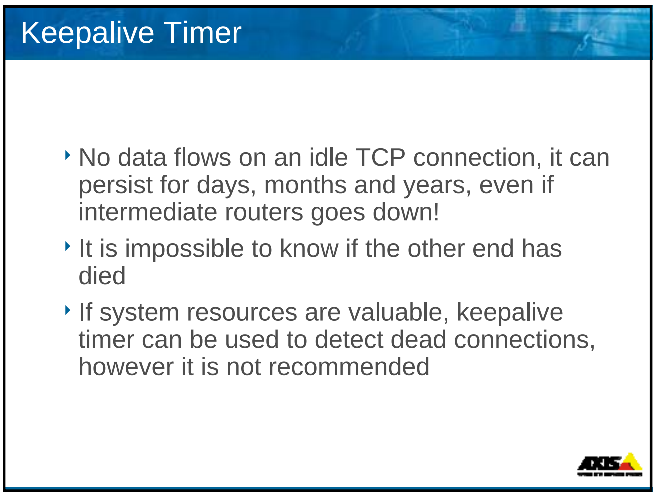
TCP Persist Timer

- If the window size is 0 and the ACK is lost, then receiver is waiting for data and sender is waiting for a non-zero window!
- Introduce a persist timer that sends window probes periodically to find out if window size has increased.
- Window probes sent every 60 seconds - TCP never gives up sending them.



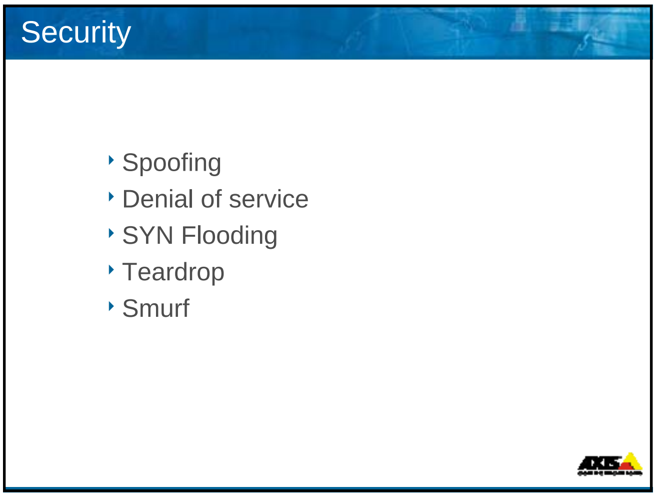
Silly Window Syndrome

- If receiver advertises a small window, then sender will send a small amount of data, which fills receivers window, ...
- Receiver must not advertise small segments
- Sender does not transmit unless:
  - Full-size segment can be sent
  - Everything can be sent



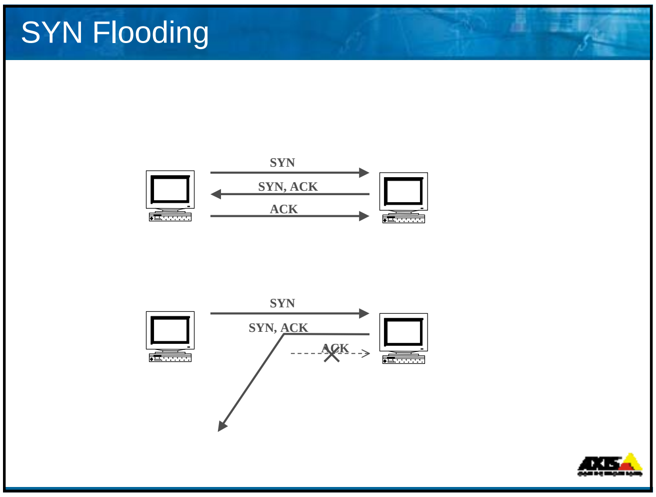
Keepalive Timer

- No data flows on an idle TCP connection, it can persist for days, months and years, even if intermediate routers goes down!
- It is impossible to know if the other end has died
- If system resources are valuable, keepalive timer can be used to detect dead connections, however it is not recommended

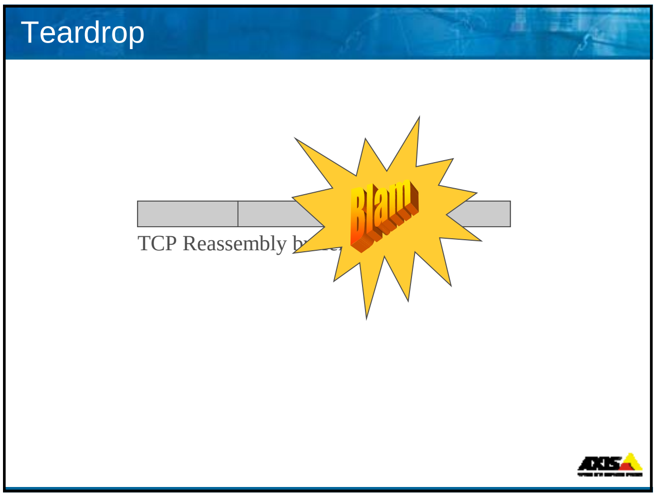


Security

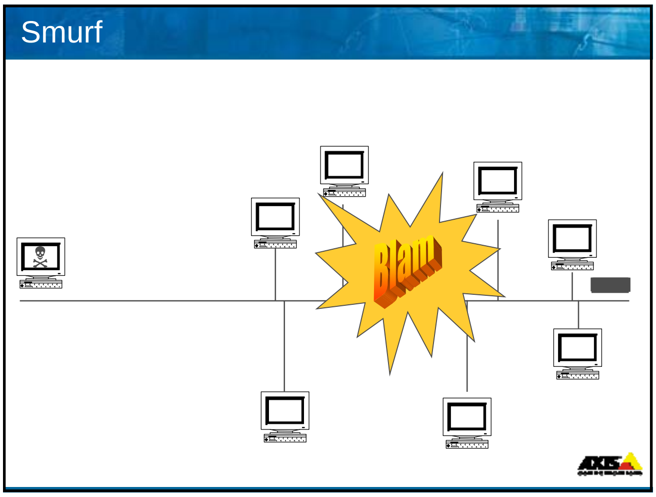
- Spoofing
- Denial of service
- SYN Flooding
- Teardrop
- Smurf



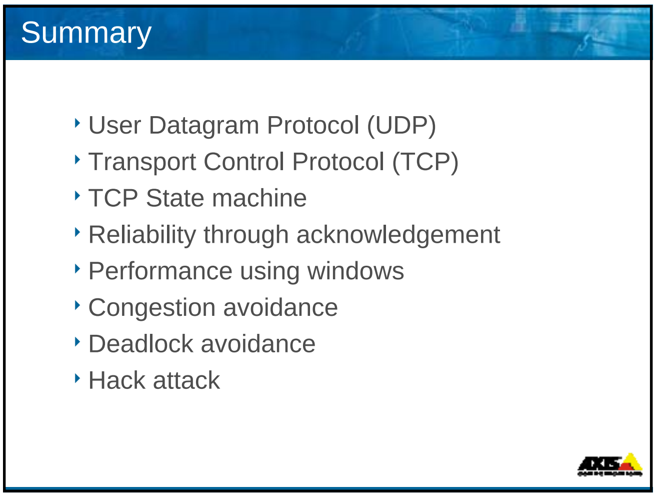
SYN Flooding



Teardrop



Smurf



Summary

- User Datagram Protocol (UDP)
- Transport Control Protocol (TCP)
- TCP State machine
- Reliability through acknowledgement
- Performance using windows
- Congestion avoidance
- Deadlock avoidance
- Hack attack