

CS 414 – Multimedia Systems Design
Lecture 23 -
Case Studies for Multimedia
Network Support (Layer 3-4)

Klara Nahrstedt
Spring 2008



Administrative

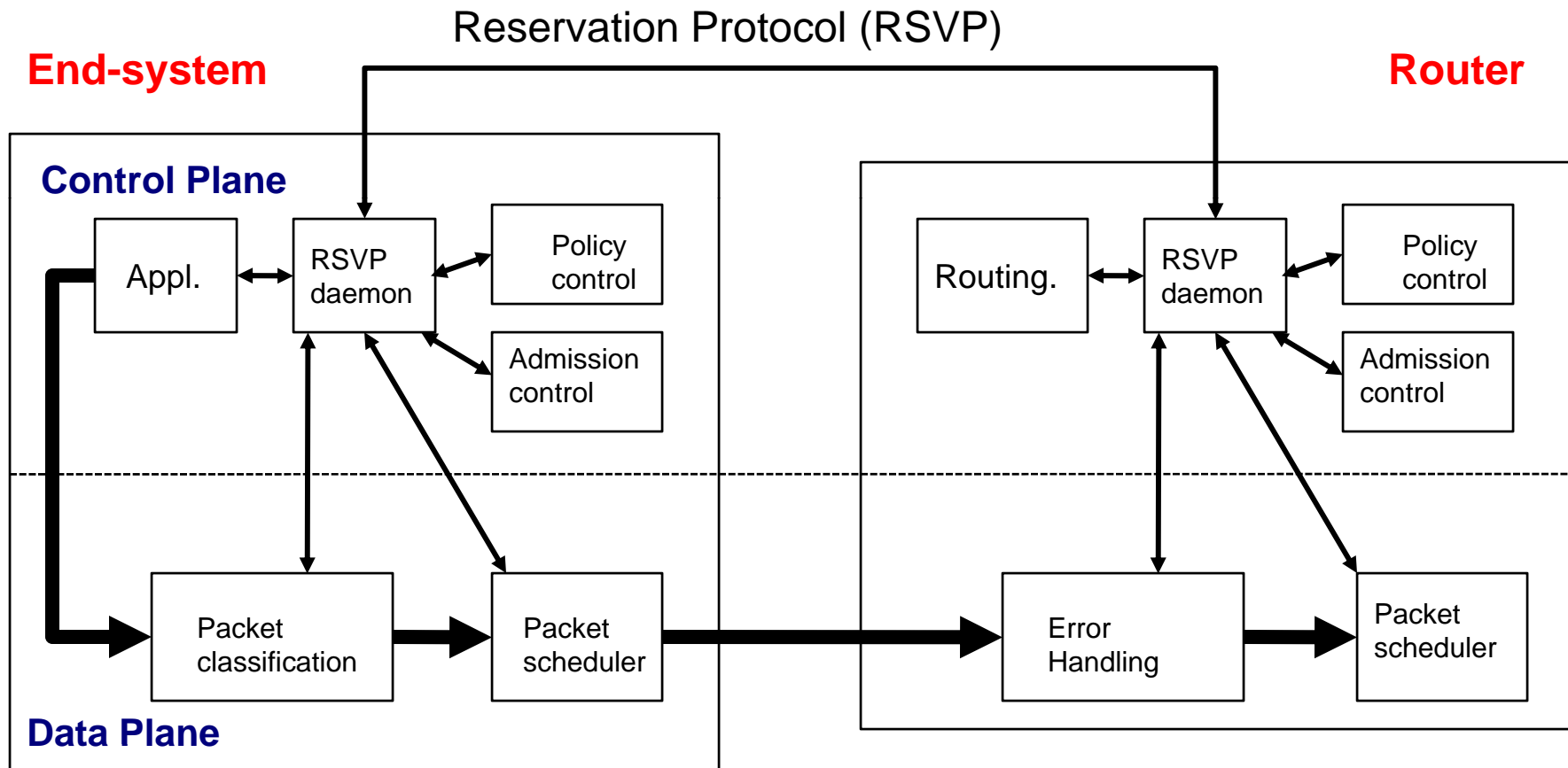
- Posted statistics of Midterm and course grading
- Regrading of Midterm until March 14
- MP3 is out – deadline April 4
 - We will have at least two discussion sections
 - One this week and one during the week after Spring break
 - If your group shrank below three members, let William and me know and we will propose merging of groups as soon as possible



Outline

- Multimedia Network Technologies at the Layer 4
 - Past/Current technologies: TCP/UDP
 - Next Generation technologies: RTP/RTCP

Integrated Services (IntServ) Architecture





Guaranteed Service

(in IntServ)

- Provides guaranteed BW and strict bounds on end-to-end queuing delay for conforming flows
- Controls max. queuing delay
- **TSpec** – describes traffic sources
 - **Bucket rate** ('r') (bytes/second)
 - **Peak rate** (p) (bytes/second)
 - **Bucket depth** (b) (bytes)
 - **Minimum policed unit** (m) (bytes) – any packet with size smaller than *m* will be counted as *m* bytes
 - **Maximum packet size** (M) (bytes) – max, packet size that can be accepted



Guaranteed Service (2)

- **Rspec**

- **Service rate** (R) (bytes/second) – service rate or BW requirement
- **Slack term** (S) (μsec) – extra amount of delay that a node may add that still meets the EED (end-to-end delay) requirement.

- This service does policing and shaping
- Resources are reserved at worst case
- For bursty traffic sources – low network utilization



Controlled Load Service

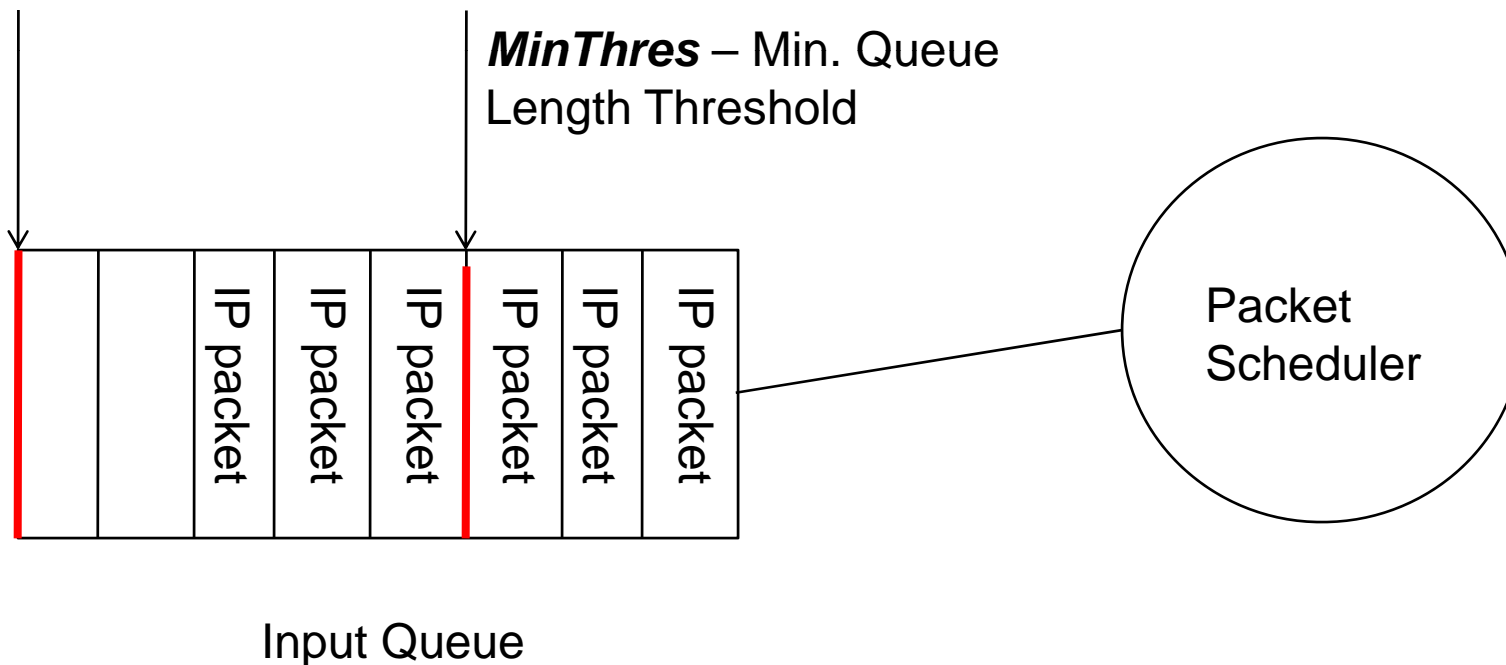
(in IntServ)

- No quantitative guarantees on delay bound or BW
- This service model allows statistical multiplexing
 - statistical guarantees
 - Very high % of transmitted packets will be successfully delivered
 - Transit queuing delay experienced by a very high % of delivered packets will not greatly exceed min. delay
- Invocation and Policing
 - Specify TSpec (average values) and do admission, reservation, policing based on average TSpec

IntServ (Error Handling - Early Congestion Avoidance)

Avr – Average Queue Length

MaxThres – Max Queue Length Threshold



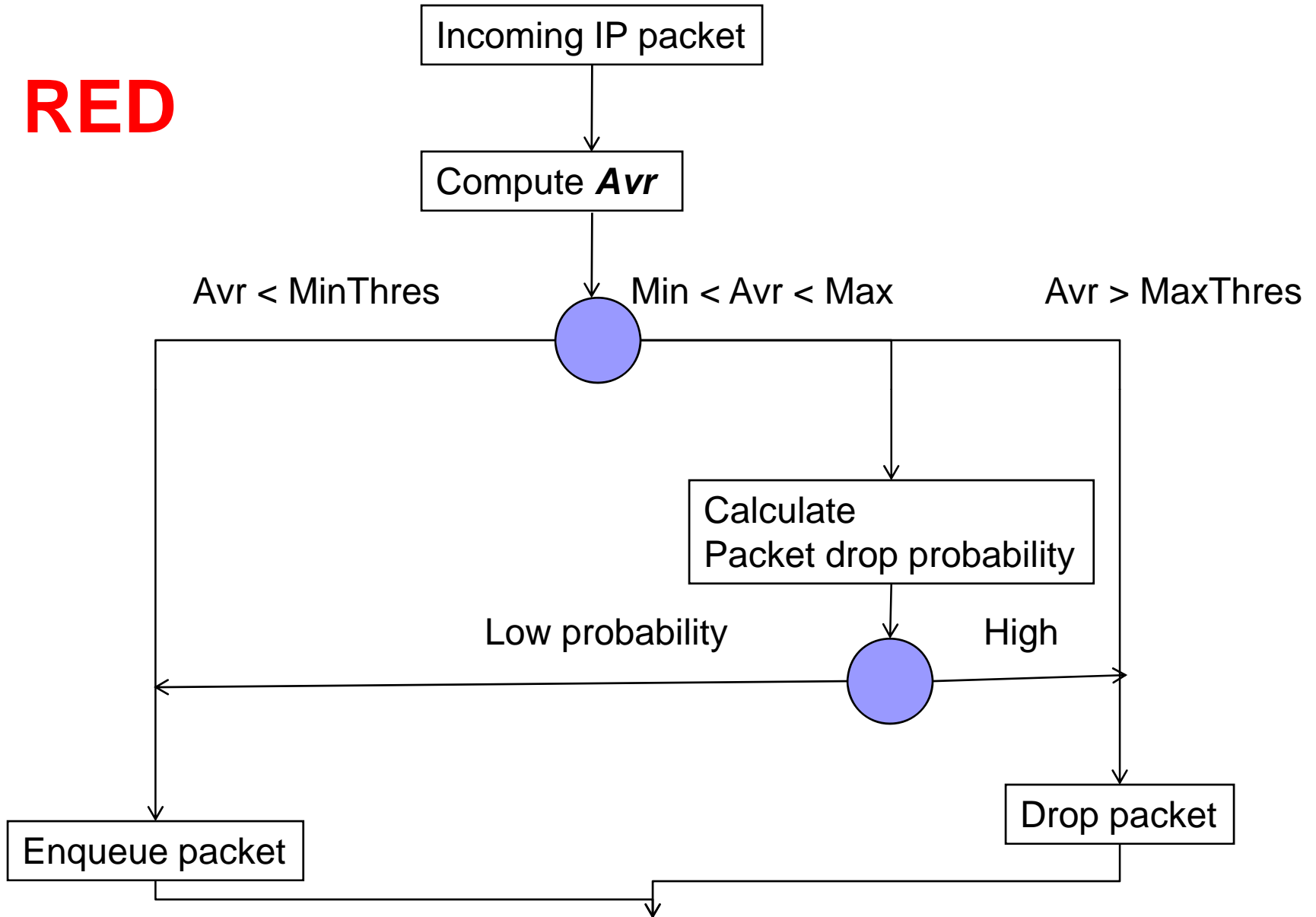


IntServ (Error Handling)

Discard Algorithms

- RED: Random Early Detection;
 - single FIFO queue is maintained for all packets and packets are dropped randomly with a given probability when the average queue length exceeds minimum threshold (MinThresh). If max. threshold (MaxThresh) is exceeded, all packets are dropped
- WRED – Weighted RED
 - Drops packets selectively based on IP precedence

RED





Packet Scheduling

(in IntServ)

■ Isolation versus Sharing

- Circuit-switched network (e.g., telephone network) – all flows are isolated, i.e., each connection has dedicated resource
- Datagram-based Internet – all resources are shared on per-packet basis without any form of isolation and protection

■ IntServ requires scheduling algorithms to support delay bounds

- Deterministic or statistical delay bounds
- Deterministic and statistical bounds reflect trade-offs between isolation and sharing



Packet Scheduling

(in IntServ)

■ Simple Priority

- Be careful – a large volume of higher-priority packets can easily starve lower-priority packets

■ Fair queuing approach

- Allocate BW proportional to active flows based on their weights

■ Deadline-based schemes

- Use EDF on packets

■ Rate-based scheduling framework

- Has two components: regulator and scheduler
- Example: token bucket with fair queuing



Transport Protocols (Layer 4)

- Existing Protocols –
 - TCP – Reliable Transport Protocol
 - UDP – Unreliable Transport Protocol
- New Protocols –
 - RTP – Real-time Transport protocol
 - RTCP – Real-time Control Protocol



TCP- Transmission Control Protocol - Features

- Serial communication path between processes exchanging a full-duplex stream of bytes
- Sequential delivery (no reordering required)
- Reliable delivery
 - Achieved through retransmission via timeouts and positive acknowledgement on receipt of information
- Flow and congestion control is based on window technique

TCP Header

Bit offset	Bits 0–3	4–7	8–15	16–31							
0	Source port			Destination port							
32	Sequence number										
64	Acknowledgment number										
96	Data offset	Reserved	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window
128	Checksum			Urgent pointer							
160	Options (optional)										
160/192 +	Data										



Flow and Congestion Control in TCP

- Slow-start algorithm – basic flow and congestion control in TCP
- The algorithm requires sender to keep **congestion window** which is the estimate of how much traffic the network can actually take
- Congestion window is managed using **two-part algorithm**:
 - Sender sends exponentially until TCP segment gets lost
 - Sender sends exponentially up to half the previous window, then window grows linearly



Techniques for Going Faster

- TCP predictions (1987) that TCP/IP cannot go faster than 10 Mbps
- Van Jacobson investigated making TCP faster
- Techniques:
 - Memory management – reduce copying
 - Interrupt handling – clocked interrupts



Techniques for Going Faster

■ Better lookup techniques

- TCP must lookup connection block for each segment received
- IP must find a route to be able send IP packet

■ Use caches of frequently used information

- Maximize hit rate, minimize search and maintenance
 - Most effective – small caches
 - Packets travel in packet rates
 - **CACHE OF 20 ROUTES SHOWED HIT RATE OF 90%**



Techniques for Going Faster

■ Lookup algorithm

- Hashing using open chaining – head of each hashed link list keeps a cache of the last accessed control block

■ Prediction

- TCP behavior is highly predictable and one can take advantage by optimizing the frequent path through TCP code at sender/receiver
- Header prediction



Sequence Numbers

- High **delay-bandwidth product** has implication on TCP window size and sequence space;
- TCP window size is 64 KB – we need possibility to negotiate the window size
- Sequencing uses wrap-around counters to put in sequence numbers
 - Sequence number space is too small
- Examples:
 - In case of **10 Mbps**, the IP packet lifetime was designed with 120 seconds and sequence space of 32 bits – takes about **1700 seconds** to send 2^{31} bytes
 - In case of **1 Gbps**, it takes **17 seconds** to send 2^{31} bytes



Conclusion

- TCP has been augmented and can achieve high performance suitable for multimedia, but one must optimize TCP for performance
 - Especially for large video streams