1 QOS Requirements and Service Level Agreements

1.1 Introduction

When sending a parcel, the sender can generally select from a range of contractual commitments from the postal courier service provider; that the parcel will arrive within two working days of being sent, for example. The commitments may include other parameters or metrics such as the number of attempts at redelivery if the first attempt is unsuccessful, and any compensation that will be owed by the courier if the parcel is late or even lost. The more competitive the market for the particular service, the more comprehensive and the tighter the commitments or service level agreements (SLAs) that are offered.

In the same way, within the networking industry the increased competition between Internet Protocol (IP) [RFC791] service providers (SPs) together with the heightened importance of IP applications to business operations has led to an increased demand and consequent supply of IP services with better defined and tighter SLAs for IP performance. These SLAs represent a contract for the delivery of the service; in this case, it is an IP transport service. The SLA requirements of a service need to be derived from the SLA requirements of the applications they are intended to support; customers utilizing the service rely on this contract to ensure that they can deliver the applications critical to their business. Hence, SLA definitions are key and it is essential they are representative of the characteristics of the IP transport service they define.

For an IP service, the service that IP traffic receives is measured using quality metrics; the most important metrics for defining IP service performance are:

- delay
- delay variation or delay-jitter
- packet loss
- throughput
- service availability
- per flow sequence preservation.

"Quality of service" or QOS (either pronounced "Q-O-S" or "kwos") implies providing a contractual commitment (SLA) for these quality metrics. This contract may be explicitly defined; it is common for an IP transport service to have such an explicit SLA, for example. Alternatively, SLAs may be implied; for example, if you upgrade from a 2 Mbps DSL Internet connection to an 8 Mbps connection then you might expect that the service that you receive improves; however, this need not necessarily be the case. In this example "2 Mbps" and "8 Mbps" define the maximum rates for the service and as DSL services are commonly delivered using contended access networks, the actual usable throughput that users experience may be less than this maximum rate. Hence, the way in which the network has been engineered to deliver the service will determine the throughput that users receive, and it is possible that even though the user's maximum rates are different, their attained usable throughput may be the same. Clearly, there is no incentive for end-users to upgrade from a 2 Mbps service to an 8 Mbps service if they do not perceive a difference between them. Hence, in reality there is an implied SLA difference between the two services even if it is not explicitly specified – that the 8 Mbps service will offer a higher attainable throughput than the 2 Mbps service.

Application and service SLA requirements are the inputs and also the qualification criteria for measuring success in a network QOS design; a network which provides a 500 ms one-way delay would clearly not be able to support a voice over IP (VoIP) service requiring a worst-case one-way delay of 200 ms. Similarly, a network that provides a one-way delay of 50 ms may be over-engineered to support this service, and over-engineering may incur unnecessary cost. In price-sensitive markets, whether customers will be prepared to pay for the facility that QOS provides may depend in part on whether they can detect the effects of QOS; SLAs can provide a means to qualify the difference between services.

Although it is common for SPs, who provide virtual private network (VPN) services to enterprise organizations, to offer an explicit SLA to their enterprise customers, it is less common within enterprise organizations to define explicitly the SLAs that they engineer their networks to support. Nonetheless, enterprise networks support business-critical applications that have bounded SLA requirements; without an understanding of these requirements, it is not possible to engineer a network to ensure that they can be adequately supported without the risk of over- or under-engineering. An understanding of application SLA requirements is therefore as important in enterprise networks as in network SP environments.

In considering SLAs and SLA metrics, because they define a service contract, as with any contract the detail of the contract definition matters. In terms of SLAs for IP service performance, it is important to understand how the SLAs are numerically defined; SLAs may be defined in absolute terms, e.g. a worst-case one-way delay of 100 ms, or may be defined statistically, e.g. a loss rate of 0.01%. In the case of the statistical definition, defining a network loss rate of 0.01% is not sufficient information on its own to be able to determine if an application or service could be supported on that network. How the loss rate is measured and calculated needs to be defined in order to understand what impact the 0.01% loss rate will have on the end applications; 1 lost packet in every one hundred packets may not have a significant impact on a VoIP call, but 10 consecutive packets dropped out of 1000 will cause a glitch in the call that is audible to the end-user.

In order to remove some of the potential ambiguity around SLA definitions, the IP Performance Metrics [IPPM] Working Group (WG) within the Internet Engineering Task Force (IETF) was tasked with defining a set of standard metrics and procedures for accurately measuring and documenting them, which can be applied to the quality, performance, and reliability of IP services. The intent of the IPPM WG was to design metrics such that they can be measured by network operators, end-users, or independent testing groups. Their aim was to define metrics that do not represent a subjective value judgment (i.e. do not define "good" or "bad"), but rather provide unbiased quantitative measures of performance. [RFC2330] defines the "Framework for IP Performance Metrics" within the IETF. It is noted, however, that the SLAs provided by network service providers to customers do not generally use the IPPM definitions; see Section 1.4 for a discussion on "marketing" versus "engineering" SLAs.

In the proceeding sections in this chapter, we consider the SLA's metrics that are important for IP service performance in more detail, review the current industry status with respect to the standardization, and support of these metrics and then describe application SLA requirements and the impacts that these metrics can have on application performance.

1.2 SLA Metrics

1.2.1 Network Delay

SLAs for network delay are generally defined in terms of one-way delay for non-adaptive (inelastic) time-critical applications such as VoIP and video, and in terms of round-trip delay or round-trip time (RTT) for adaptive (elastic) applications, such as those which use the Transmission Control Protocol (TCP) [RFC793].

One-way delay characterizes the time difference between the reception of an IP packet at a defined network ingress point and its transmission at a defined network egress point. A metric for measuring one-way delay has been defined by [RFC2679] in the IETF.

RTT characterizes the time difference between the transmission of an IP packet at a point, toward a destination, and the subsequent receipt of the corresponding reply packet from that destination, excluding end-system processing delays. A metric for measuring RTT has been defined by [RFC2681] in the IETF.

Whether considering one-way delay or round-trip delay, the delays induced in a network are made up of the four following components.

1.2.1.1 Propagation Delay

Propagation delay is the time taken for a single bit to travel from the output port on a router across a link to another router. This is constrained by the speed of light in the transmission medium and hence depends both upon the distance of the link and upon the physical media used. The total propagation delay on a path consisting of a number of links is the sum of the propagation delays of the constituent links. Propagation delay is around 4 ms per 1000 km through coaxial cable and around 5 ms per 1000 km for optical fiber (allowing for repeaters).

In practice, network links never follow the geographical shortest path between the points they connect, hence the link distance, and associated propagation delay, can be estimated as follows:

- Determine the "as the crow flies" geographical distance D between the two end points.
- Obviously, the link distance must be longer than the distance as the crow flies. The route length R can be estimated from D, for example, using the calculation from International Telecommunications Union (ITU) recommendation [G.826], which is summarized in the following table.

D	R
$D < 1000$ km	$R = 1.5 * D$
1000 km $\leqslant D \leqslant$ 1200 km	$R = 1500$ km
$D > 1200$ km	$R = 1.25 * D$

The only way of controlling the propagation delay of a link is to control the physical link routing, which could be controlled at layer 2 or layer 3 of the Open Systems Interconnection (OSI) 7 layer Reference Model. If propagation delays for a link are too large, it may be that the link routing in an underlying layer 2 network is longer than it needs to be, and may be reduced by rerouting the link. Alternatively, a change to the network topology, by the addition of a more direct link for example, may reduce the propagation delay on a path.

1.2.1.2 Switching Delay

The switching or processing delay incurred at a router is the time difference between receiving a packet on an incoming router interface and the enqueuing of the packet in the scheduler of its outbound interface. Switching delays on high-performance routers can generally be considered negligible: for backbone routers, where switching is typically implemented in hardware, switching delays are typically in the order of 10–20 μs per packet; even for software-based router implementations, typical switching delays should only be 2–3 ms.

Little can be done to control switching delays without changing router software or hardware; however, as switching delays are generally a minor proportion of the end-to-end delay, this will not normally be justified.

1.2.1.3 Scheduling Delay

Scheduling (or queuing) delay is defined as the time difference between the enqueuing of a packet on the outbound interface scheduler, and the start of clocking the packet onto the outbound link. This is a function of the scheduling algorithm used and of the scheduler queue utilization, which is in turn a function of the queue capacity and the offered traffic load and profile.

Scheduling delays are controlled by managing the traffic load and by applying appropriate queuing and scheduling mechanisms (see Chapter 2, Section 2.2.4.1).

1.2.1.4 Serialization Delay

Serialization delay is the time taken to clock a packet onto a link and is dependent upon the link speed and the packet size. Serialization

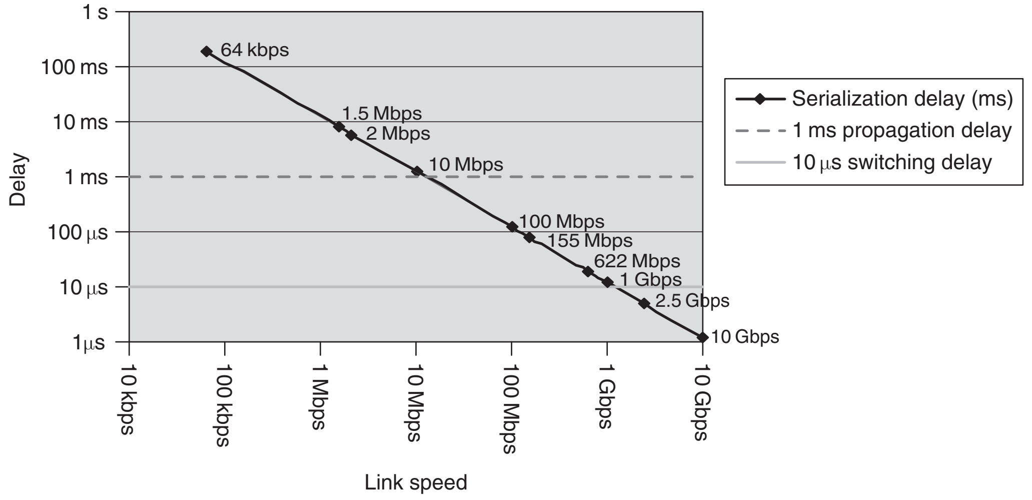

Figure 1.1 Serialization delay for 1500 byte packet

delay is proportional to packet size and inversely proportional to link speed:

$$\text{serialization\_delay} = \frac{\text{packet\_size}}{\text{link\_speed}}$$

Serialization delay can generally be considered negligible at link speeds above 155 Mbps (e.g. STM-1/OC3) such as backbone links, but can be significant on low-speed links. The serialization delay for a 1500-byte packet at link speeds from 64 kbps to 10 Gbps is shown in Figure 1.1, together with a line plotting indicative switching delay and a line showing a propagation delay of 1 ms (e.g. a link distance of ~130 km).

Serialization delay clearly is more significant component of delay for lower-speed links. Serialization delay is a physical constraint and hence there is no way of controlling serialization delay other than changing the link speed.

1.2.2 Delay-jitter

Delay-jitter characterizes the variation of network delay. Jitter is generally considered to be the variation of the one-way delay for two consecutive packets, as defined by [RFC3393] in the IETF. In practice, however, jitter can also be measured as the variation of delay with respect to some reference metric, such as average delay or minimum delay. It is fundamental that jitter relates to one-way delay; the notion of round-trip time jitter does not make sense.

Jitter is caused by the variation in the components of network delay previously described in Section 1.2.1:

- *Propagation delay*. Propagation delay can vary as network topology changes, when a link fails, for example, or when the topology of a lower layer network (e.g. SDH/SONET) changes, causing a sudden peak of jitter.
- *Switching delay*. Switching delay can vary as some packets might require more processing than others might. This effect may be perceptible in software-based router implementations but is becoming less of a consideration as routers implement packet switching in hardware resulting in more consistent switching delay characteristics.
- *Scheduling delay*. Variation in scheduling delay is caused as schedulers' queues oscillate between empty and full.
- *Serialization delay*. Serialization delay is a constant and as such should not contribute to jitter directly. If during a network failure, however, traffic is rerouted over a link with a different speed, then the serialization delay will change as a result of the failure and the change in serialization delay may contribute to jitter.

Some applications, such as those which use TCP, are generally not susceptible to jitter. Applications that are susceptible to jitter use dejitter buffers in order to remove delay variation by turning variable network delays into constant delays at the destination end-systems. Considerations on de-jitter buffers and their tuning are discussed in more detail in Section 1.3.1.2.

1.2.3 Packet Loss

Packet loss characterizes the packet drops that occur between a defined network ingress point and a defined network egress point. A packet sent from a network ingress point is considered lost if it does not arrive at a specified network egress point within a defined time period.

A metric for measuring the one-way packet loss rate (PLR) has been defined by [RFC2680] in the IETF.

One-way loss is measured rather than round-trip loss because the paths between a source and destination may be asymmetrical; that is, the path routing or path characteristics from a source to a destination may be different from the path routing or characteristics from the destination back to the source. Round-trip loss can be estimated by measuring the loss on each path independently.

In addition to the measured loss rate, in some applications the loss pattern or loss distribution is a key parameter that can impact the performance observed by the end-users; the same loss rate can result in significantly different perceptions of performance given two different loss distributions. Consequently, [RFC3357] introduces some additional metrics, which describe loss patterns:

- "loss period" defines the frequency and length of loss (loss burst) once it starts
- "loss distance" defines the spacing between the loss periods.

Packet loss can be caused by a number of factors:

- *Congestion*. When congestion occurs, queues build up and packets are dropped. Loss due to congestion is controlled by managing the traffic load and by applying appropriate queuing and scheduling mechanisms (see Chapter 2, Section 2.2.4).
- *Lower layer errors*. Physical layer bit errors, which may be due to noise or attenuation in the transmission channel, may cause packets to be dropped. Most link layer technologies and IP transport

protocols, such as the User Datagram Protocol (UDP) [RFC768], have a cyclic redundancy check (CRC) or parity checksum to detect bit errors; when bit errors occur and the checksum is incorrect, the impacted frames will be dropped. Hence, for packets traversing networks with such capabilities, bit errors will normally result in packet loss, i.e. each packet will either arrive correct or not at all, although there are a few noted exceptions to this (see Section 1.3.2.1.3). In practice, actual bit error rates (BER, also referred to as the bit error ratio) vary widely depending upon the underlying layer 1 or layer 2 technologies used, which is different for different parts of the network:

- Fiber-based optical links may support bit error rates as low as to $1 * 10^{-13}$
- Synchronous Digital Hierarchy (SDH) or Synchronous Optical Network (SONET) services typically offer BER of $1 * 10^{-12}$
- Typical E1/T1 leased line services support BER of $1 * 10^{-9}$
- The Institute of Electrical and Electronics Engineers (IEEE) standard for local and metropolitan area networks [802-2001] specifies a maximum BER of $1 * 10^{-8}$
- Typical Asynchronous Digital Subscriber Line (ADSL) services support BER of $1 * 10^{-7}$
- Satellite services typically support BER of $1 * 10^{-6}$

For link layer technologies that are generally prone to high error rates, it is usual to support some link layer reliability mechanisms, such as Forward Error Correction (FEC), in order to recover from some bit error cases. If, however, the underlying layer 1 or layer 2 technologies cannot provide the BERs necessary to support the packet loss rates (PLRs) required by IP applications, then error correction or concealment techniques need to be used either by higher layer protocols or by the application, or alternate layer 1 or layer 2 technologies are needed.

- *Network element failures*. Network element failures may cause packets to be dropped until connectively is restored around the failed network element. The resulting loss period depends upon the underlying network technologies that are used.

With a "plain" IP (i.e. non-MPLS) deployment, after a network element failure, even if there is an alternative path around the failure, there will be a loss of connectivity which causes packet loss until the interior gateway routing protocol (IGP) converges. In well-designed networks, the IGP convergence time completes in a few hundred milliseconds [FRANCOIS]. If there is not an alternative path available then the loss of connectivity will persist until the failure is repaired. While such outages could be accounted for by the defined loss rate for the service, they are most commonly accounted for in the defined availability for the service (see Section 1.2.6).

Where an alternate path exists, the loss of connectivity following network element failures can be significantly reduced through the use of technologies such as MPLS Traffic Engineering (TE) Fast Reroute (FRR) [RFC4090] or IP Fast Reroute (IPFRR), which are local protection techniques that enable connectivity to be rapidly restored around link and node failures, typically within 50 ms. Equivalent techniques may be employed at layer 2, such as Automatic Protection Switching (APS) for SONET and Multiplex Section Protection (MSP) for SDH.

- *Loss in application end-systems*. Loss in application end-systems can happen due to overflows and underflows in the receiving buffer. An overflow is where the buffer is already full and another packet arrives, which cannot therefore be enqueued in the buffer; overflows can potentially impact all types of applications. An underflow typically only impacts real-time applications, such as VoIP and video, and is where the buffer is empty when the codec needs to play out a sample, and is effectively realized as a "lost" packet.

  Loss due to buffer underflows and overflows can be prevented through careful design both of the network and the application end-systems.

Depending upon the transport protocol or application, there are potentially a number of techniques that can be employed to protect against packet loss including error correction, error concealment, redundant transmission and retransmission.

1.2.4 Bandwidth and Throughput

IP services are commonly sold with a defined "bandwidth," where the bandwidth often reflects the layer 2 access link capacity provisioned for the service; however, when used in the context of networking the term "bandwidth" – which was originally used to describe a range of electromagnetic frequencies – can potentially have a number of different meanings with respect to the capacity of a link, network or service to transport traffic and data. Hence, to avoid confusion we define some more specific terms:

- *Link capacity*. The capacity of a link is a measure of how many bits per second that link can transport; link capacity needs to be considered both at layer 2 and at layer 3.
  - The capacity of a link is normally constant at layer 2 and is a function of the capacity of the physical media (i.e. the layer 1 capacity) and particular layer 2 encoding used. Some media, however, such as ADSL 2/2+ are rate-adaptive, and hence the layer 1 capacity can vary with noise and interference.
  - Link capacity at layer 3 (i.e. the IP link capacity) is a function of the link capacity at layer 2, the layer 2 encapsulation used and the layer 3 packet sizes. The IP link capacity can be derived for IP packets of a specified size, from the available layer 2 link capacity in bits per second, where only those bits of the IP packet are counted. The effect of layer 2 overheads on SLA definitions is discussed in more detail in Section 1.2.4.1.

  Link capacity is also referred to as link bandwidth or link speed.

- *Class capacity*. Where QOS mechanisms are used, an aggregate traffic stream may be classified into a number of constituent classes, and different QOS assurances may be provided to different classes within the aggregate. Where a class has a defined minimum bandwidth assurance, this is referred to as the class capacity, and may also be known as the class bandwidth.

- *Path capacity*. Path capacity is the minimum link capacity on a path between a defined network ingress point and a defined network egress point, consisting of a number of links interconnected by a number of nodes or routers. [CHIMENTO] also provides definitions for link and path capacity. Path capacity may also be referred to as the path bandwidth.

- *Bulk Transport Capacity*. The Bulk Transport Capacity (BTC) is a measure of the attainable user data throughput between a source and a destination; [RFC3148] specifies a framework for Defining Empirical Bulk Transfer Capacity Metrics. BTC is effectively a measure of the long-term average data throughput rate (e.g. in bits per second) a single congestion-aware transport layer connection could achieve over the path from source to destination. "Congestion aware" in this context refers to a transport layer technology that adapts its rate of sending, depending upon what is actually received, in order to try to maximize throughout; a TCP session is an example of such a congestion-aware transport layer connection. BTC is clearly limited by the path capacity, but is also impacted by a number of other factors such as packet loss and RTT (see Section 1.3.3.1), hence it is important to note that the BTC may be significantly lower than the link capacity specified in the SLA. BTC is a representation of the "goodput" available to a user, where the goodput represents the usable portion of the attainable throughput between a source and destination.

  BTC is not applicable to non congestion-aware, i.e. non-adaptive or inelastic, services; for such services, their attainable throughput may not be a meaningful metric, but nonetheless may be derived from the path capacity and the loss rate commitments for the service.

Hence, it is clear that the throughput attained for a service may not be the same as the defined "bandwidth." The following sections consider additional factors that further complicate the relationship between "bandwidth" and attained throughput.

1.2.4.1 Layer 2 Overheads

When considering the capacity of a service the available IP capacity depends upon the layer 2 media, the layer 2 encapsulation used and

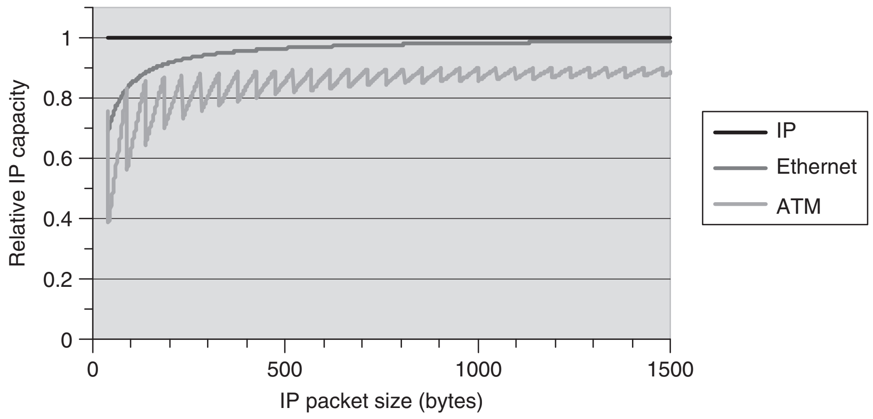

Figure 1.2 Relative IP capacity for different layer 2 media

upon the layer 3 packet sizes. Different layer 2 encapsulations add different sized headers and trailers to each packet; the headers and trailers are an overhead from the perspective of IP services, in that they use available layer 2 capacity, which is therefore not available at layer 3. As the layer 2 headers and trailers are added to each IP packet, the amount of layer 2 overhead incurred, and hence the IP capacity available, is dependent upon the IP packet size. Figure 1.2 shows the relative IP capacity for Ethernet and ATM connections, and how this relative capacity varies with IP packet sizes.

As can be seen from Figure 1.2, the available IP capacity can vary significantly depending upon the overall layer 2 overhead. For Ethernet, the layer 2 overhead in bytes per IP packet is constant, irrespective of the packet size; hence the layer 2 overhead reduces relative to the available IP capacity as the IP packet size increases. This is not the case for ATM, where an IP packet is segmented into cells and the per cell overhead or "cell tax" depends upon the number of cells, which in turn depends on the packet size. Hence, although the trend in the layer 2 overhead is to reduce relative to the available IP capacity as the IP packet size increases, a one-byte increase in packet size can result in an additional ATM cell, which results in an increase in the relative overhead; hence the saw tooth IP capacity characteristic for ATM in Figure 1.2.

Some services use traffic shapers applied to the access links in order to reduce the available capacity of the link; however, shapers, policers and schedulers can also exhibit very different behaviors, depending on whether they account for bandwidths in terms of layer 3 packet sizes or whether they also include all layer 2 overheads, or even account for something in between the two.

We discuss scheduling in detail in Chapter 2, Section 2.2.4.1; however, prior to that, consider for example, a simple two-queue (where a queue is ostensibly a class) scheduler with per-queue minimum bandwidth assurances defined at layer 3 of X = Y = 50%. With IP packet sizes of 100 bytes for queue X and 1000 bytes for queue Y, and assuming a layer 2 overhead of 26 bytes per packet (as is the case with Ethernet v2), the measured bandwidth ratio X:Y at layer 3 is (10 * 100):(1 * 1000) = 50:50, whereas the ratio measured at layer 2 = (10 * 126):(1 * 1026) = ~55:45. Conversely, assuming the same packet sizes and overhead but with per-queue minimum bandwidth assurances of X = Y = 50% defined at layer 2, the resulting bandwidth ratio at layer 3 = (100 * 1026):(1000 * 126) = ~45:55.

In some cases, there are constraints imposed by the underlying layer 1 and layer 2 technologies, which naturally define the overheads that are taken into account in a particular SLA definition. In other cases there may be no definitive answer as to whether layer 2 overheads should be taken into account:

- Accounting for all layer 2 overheads in actual router implementations can be difficult when, for example, layer 2 fragmentation mechanisms (see Chapter 2, Section 2.2.5) insert additional bytes after a packet has been enqueued.
- Some service SLAs are defined excluding layer 2 overheads; while others to take layer 2 overheads into account; there is no de facto industry approach.
- The IETF's Integrated Services (Intserv) and Differentiated Services (Diffserv) architectures (see Chapter 2, Sections 2.3.3 and 2.3.4) do not discuss or define the accounting of layer 2 overheads.

- There is variation in the overheads accounted for by different vendors' scheduling, shaping, and queuing implementations; in some vendors' implementations the overhead accounting that is taken into account by a QOS policy can be configured.

Whichever approach is adopted, the SLA specification must clearly define which overheads are taken into account and to which layer the bandwidth assurances apply. This has a consequent impact on the overheads that QOS functions such as scheduling, shaping, or policing need to take into account.

1.2.4.2 VPN Hose and Pipe Models

Consider a network connecting four sites (#1, #2, #3 and #4); this could be a layer 2 virtual private network (VPN) offered by a service provider using a technology such as leased lines, Frame-relay or ATM, for example. Whichever underlying technology is used the sites could be interconnected in a hub and spoke arrangement with spoke sites connected back to a hub site using leased lines or virtual circuits (VCs), or they could be interconnected with a full mesh of leased lines or VCs; both options are shown in Figure 1.3.[1]

In both cases shown in Figure 1.3, each leased line or VC may have a defined SLA commitment; this type of point-to-point bandwidth commitment was first termed a "pipe" by [DUFFIELD]; these point-to-point commitments provide isolation between the performance of

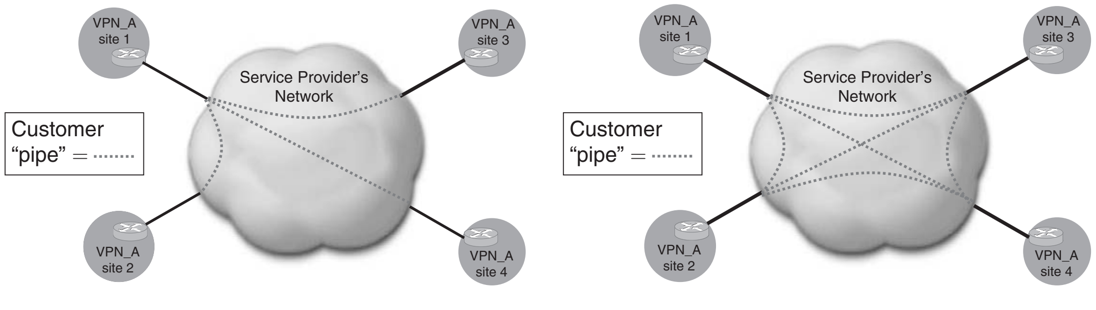
VPN_A site 1
VPN_A site 3
Service Provider's Network
Customer "pipe" = ········
VPN_A site 2
VPN_A site 4
VPN_A site 1
VPN_A site 3
Service Provider's Network
Customer "pipe" = ········
VPN_A site 2
VPN_A site 4

Figure 1.3 Hub and spoke (left) and full mesh (right) VPNs using the "pipe model"

each "pipe." The use of the "pipe model" is obvious for point-to-point services such as leased lines, Frame relay, ATM or layer 2 "pseudo wires" as defined by the Pseudo Wire Emulation Edge-to-Edge Working Group [PWE3] within the IETF. When layer 3 services are built on top of such point-to-point pipes, however, as the number of sites within a VPN increases, provisioning such point-to-point commitments can become cumbersome. For example, a full mesh between n sites requires $n(n-1)/2$ connections, e.g. 100 sites would require 100 * 99/2 = 4950 such pipes. In addition, such point-to-point commitments can be inefficient with respect to the use of provisioned capacity. For example, site #1 may have 1 Mbps VCs provisioned to each of sites #2, #3, and #4 (i.e. 3 Mbps in total), yet if the VC to site #2 is not busy, this unused capacity to/from site #2 cannot be re-used for traffic between site #1 and sites #3 or #4, i.e. up to 1 Mbps of capacity to/from site #1 would go idle.

VPNs built using IP or MPLS technology, e.g. BGP MPLS VPNs as per [RFC4364], can implicitly provide "any-to-any" connectivity between the sites within the VPN; however, this gives rise to the question of how to define SLAs between sites within VPNs that provide multipoint-to-multipoint connectivity, when you do not have a corresponding pipe (or its SLA assurances) between those sites? [DUFFIELD] addressed this by defining the "hose model" for multipoint-to-multipoint VPN services. With the hose model, rather than defining SLAs on a point-to-point basis between pairs of sites, the SLAs are defined in terms of a "hose" from each site to and from the VPN provider network. From a capacity perspective, the "hose" for each site is defined in terms of the ingress committed rate (ICR) to the provider and the egress committed rate (ECR) from the provider, as shown in Figure 1.4.

Traffic between two sites that is within the ICR contract at the source site, and within the ECR contract at the destination site is assured end-to-end. ICR/ECR could be defined with a single class per site, or in the context of a Diffserv enabled service, could be offered on a per class per site basis.

Hose model SLAs can provide the benefits of statistical multiplexing, where pipe model SLAs cannot. For example, if site #1 has an

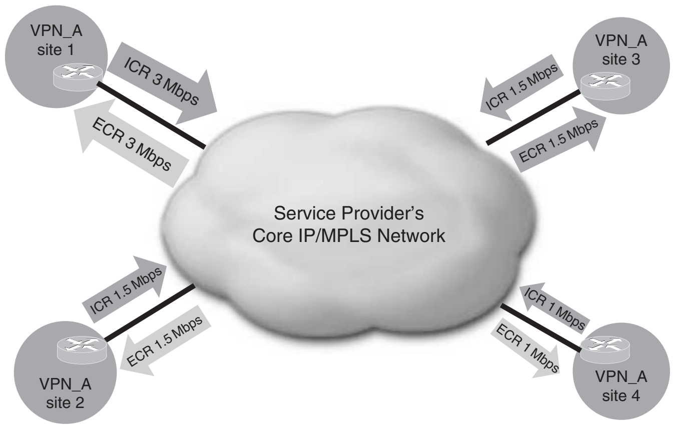

Figure 1.4 Any-to-any VPNs using the "hose model"

ICR and ECR of 3 Mbps, it could use that capacity to communicate with any of sites #2, #3, and #4, i.e. if there were no traffic to site #2, the unused capacity to/from site #2 could potentially be re-used for traffic between site #1 and sites #3 or #4. A consequence of this is that hose model SLAs also need to make provision for mediation between the ICR and ECR between different sites. For example, the ICR for site #1 may be 3 Mbps; however, the attainable capacity to site #4 will be limited by the ECR of site #4, which is 1 Mbps. In addition, hose model SLAs need to take into account cases where the loss of attainable throughput is due to customer-based traffic aggregation. For example, if sites #2, #3 and #4 all attempt to send traffic at their full ICRs (which totals 4 Mbps) to site #1, their aggregate attainable capacity will be limited by the ECR of site #1, which is only 3 Mbps.

1.2.5 Per Flow Sequence Preservation

IP does not guarantee that packets are delivered in the order in which they were sent. As defined in the IETF by [RFC4737], if the packets in a flow were numbered sequentially in the order in which they were sent, a packet that arrived with a sequence number smaller than that of their predecessor would be defined as out-of-order, or re-ordered. For example, if packets in a sequentially number stream were received in the order 1, 2, 3, 4, 7, 5, 6, 8, 9, 10 then packets numbered 5 and 6 would have been re-ordered. The simplest metric by which to measure the magnitude of re-ordering is as a re-ordering ratio, which is the ratio of re-ordered packets that arrived, relative to the total number of packets received. A number of other metrics for quantifying the magnitude of re-ordering are defined in [RFC4737].

Due to the adverse impact that packet re-ordering can have on the performance of some applications, it is accepted best practice in IP network design to prevent packet re-ordering within a flow, although it is not yet a universal component of IP service SLA commitments. There are two key design best practices in order to prevent packet re-ordering within a flow:

- It is important that any IP load balancing across multiple paths within the network is performed on a per flow level rather than on a per packet level such that all packets within a flow follow the same path. This load balancing is performed by Equal Cost Multipath (ECMP) algorithms, where there are multiple IGP paths with the same cost. ECMP algorithms commonly perform a hash function to determine which of the paths a packet will take; where the hash function uses the 5-tuple of IP protocol, source IP address, destination IP address, source UDP/TCP port, and destination UDP/TCP as inputs, and results in packets within a single flow consistently hashing to the same path.
- QOS designs and scheduling algorithms must ensure that packets from the same flow are always serviced in the same order and from the same queue; this is a fundamental principle of both the Integrated Services and Differentiated Services QOS architectures, which are described in Chapter 2.

Re-ordering of packets within a flow is bad practice and designs or implementations, which result in such re-ordering, should be considered broken!

1.2.6 Availability

Availability for IP services is generally defined in one of two ways: either as network availability or as service availability.

1.2.6.1 Network Availability

Network availability (sometimes referred to as connectivity) is defined as the fraction of time that network connectivity is available between a specified network ingress point and a specified network egress point. Availability can be unidirectional or bidirectional; bidirectional connectivity is what matters to the vast majority of IP applications, i.e. that a source can send a packet to a destination that elicits a response which is received by the source. A metric for measuring connectivity has been defined by [RFC2678] in the IETF.

Network availability needs to take into account unavailability due to planned outages, caused by scheduled network maintenance for example, as well as outages due to network failures. The unavailability that results from network failures depends upon the underlying network technologies that are used, as discussed in Section 1.2.3.

The availability of the network can be estimated by calculating the availability of each individual network element and then combining the availabilities in series or in parallel as appropriate using the following formulae.

Component availability

The availability (A) of an individual component is the proportion of the time for which the device is working:

$$A = \frac{\text{time\_working}}{\text{total\_time}} = \frac{MTBF}{MTBF + MTTR}$$

Where:
$MTBF$ = mean time between failures
$MTTR$ = mean time to restore

Component unavailability

The unavailability (U) of an individual component is the proportion of the time for which the device is not working:

$$U = \frac{\text{time\_not\_working}}{\text{total\_time}} = \frac{MTTR}{MTBF + MTTR} = 1 - A$$

Availability of components in series

The availability of components ($a,b,c, \ldots$) in series (A_s) is given by:

$$A_s = [A(a) \times A(b) \times A(c) \times \cdots]$$

Availability of components in parallel

The availability of components ($a,b,c, \ldots$) in parallel (A_p) is given by:

$$A_p = [1 - (U(a) \times U(b) \times U(c) \times \cdots)]$$

For most applications, however, simply having connectivity is not enough. For VoIP for example, it is of little practical use if there is connectivity between two VoIP end-systems but the VoIP packets arrive so delayed that speech between the two calling parties becomes unintelligible, hence service availability is often a more meaningful metric.

1.2.6.2 Service Availability

Service availability is defined as the fraction of time the service is available between a specified ingress point and a specified egress point within the bounds of the other defined SLA metrics for the service, e.g. delay, jitter, and loss. Service availability can be defined in one of two ways: either it can be defined independently of network availability, in which case the service availability cannot exceed the network availability, or it can be defined as being applicable only when the network is considered available. Service availability may encompass application performance as well as network performance. For instance, the service availability may comprise hostname resolution (DNS server) and transaction time, thereby depending on network delay and web server performance.

There may be overlap between the definition of network or service availability and the definition of other SLA parameters. For example, consider two traffic classes, A and B, where Class A supports a tighter delay SLA, which is specified with a 90th percentile (P90) delay for Class A packets of 10 ms – meaning that 99 packets out of 100 were delivered within this delay bound – and a P99 delay of 15 ms, while Class B has a P75 delay of 10 ms with a P99 delay of 30 ms. These SLAs could be expressed by a smaller delay bound for A than B but with the same availability, e.g. Class A has a delay of 15 ms with 99% availability, while Class B has a delay of 30 ms with 99% availability. Alternatively, this SLA could be expressed by the same delay bound but with a higher availability for A than for B, e.g. Class A has a delay of 10 ms with 90% availability, while Class B has a delay of 10 ms with 75% availability.

1.2.7 Quality of Experience

In addition to the metrics already described in this section, which define the characteristics of the network, there are additional metrics, which aim to quantify the performance experienced by the applications using the network. These metrics define the perception of application performance, experienced from the perspective of the end-users, which is also known as the user "quality of experience" (QOE).

For IP-based voice and video applications the QOE is a compound metric dependent upon the quality of the encoder used, the quality of the service delivered by the IP network, and the quality of the decoder used. As such, QOE targets do not directly define the delay, jitter, loss etc., characteristics that a network should provide, but rather for a specified application, using a defined encoder/decoder, the network characteristics may be implied given a particular QOE target.

QOE metrics can be measured subjectively or objectively. Subjective measures rely upon end-user feedback of their perception of the quality of the service. Objective measures use measurements of characteristics of the received stream, and possibly also of the transmitted stream, in order to infer the subjective quality that would be experienced by the end-user.

There are QOE metrics defined for voice, video and on-line gaming applications:

1.2.7.1 Voice

- *Subjective measures*. The Mean Opinion Score (MOS) is a well-established scheme, which provides a numeric measure of the quality of a voice call at the destination. MOS is a formally tested subjective measure, which is defined by the ITU [P.800] and is determined using a number of human listeners participating in a set of standard tests, subjectively scoring the quality of test sentences read aloud over the communications medium being tested using the scale: excellent (5), good (4), fair (3), poor (2) and bad (1). The MOS for the medium under test is calculated by taking the arithmetic mean of all the individual scores. A typical Public Switched Telephony (PSTN) voice service has MOS of 4.3, while mobile telephone services typically have a MOS of between 2.9 and 4.1.
- *Objective measures*. There are several recommendations provided by the ITU, which provide methods for objective voice quality monitoring, and that can also be used to estimate the MOS. These schemes rely on characteristics of the received stream only.
  - ITU P.862 [P.862] defines the Perceptual Evaluation of Speech Quality (PESQ, pronounced "pesk"), and is a full reference (where full information about both the transmitted and received audio signals are available when the audio quality is determined) objective method for predicting the subjective MOS quality of telephony services.
  - ITU G.107 [G.107] defines the so-called "E model" which uses a number of transmission level parameters to rate the quality of a transmission system, in order to assess the effects on telephony services. The primary output from the E model is the "Rating Factor," R, which can be transformed to give estimates of the MOS (an objective MOS score) for calls which use that transmission service.

1.2.7.2 Video

- *Subjective measures*. The main concepts behind the subjective measurement of video quality are the same as for MOS for voice. The most established video subjective testing scheme in the broadcasting world is the Double Stimulus Continuous Quality Scale (DSCQS) method defined in ITU specification [BT.500]. An alternative methodology that The European Broadcasting Union (EBU) [EBU] has defined is called the SAMVIQ Subjective Assessment Methodology for Video Quality [SAMVIQ].
- *Objective measures*. The most established objective measure of video quality is defined in ITU-T standard J.144 [J.144]. This provides guidelines on perceptual video quality measurement for use in digital cable television applications when the full reference video signal is available, i.e. both the transmitted and received video signals are available for comparison when the video quality is determined. It is still a subject of study as to whether reduced reference (where only partial information about transmitted video signal and full information about the received video signal is available when the video quality is determined) or no reference (where only the received video signal is available when the video quality is determined) can be used to accurately infer subjective video quality, i.e. to correctly suggest that the video quality is bad when, and only when, the end viewer also thinks it is bad.

1.2.7.3 On-line Gaming

[DICK] define a MOS metric to classify the player's perceived game quality.

1.3 Application SLA Requirements

Different applications have different SLA requirements; the impact that different network services with different SLAs have on an application is dependent upon the specific application:

- Excessive packet loss or delay may make it difficult to support real-time applications although the precise threshold of "excessive" depends on the particular application.

- The larger the value of packet loss or network delay, the more difficult it is for transport-layer protocols to sustain high bandwidths.

To appreciate these impacts, there is a minimum level of understanding required of how the applications, and the protocols they use, behave as networks SLA characteristics change. It is only through such an understanding that it is possible to engineer the network, using QOS if necessary, to ensure that the specific SLA requirements that applications require can be adequately supported, without over-engineering. Hence, this section aims to provide that minimum level of understanding, together with providing references to further detail on the application and protocol behaviors.

Additionally, by having an understanding of how applications and the protocols behave as networks SLA characteristics change, it is possible to understand where deploying QOS mechanisms may not be sufficient to be able to meet the application SLA requirements and application or network re-engineering may be required instead.

Conversely, to understand the impact that an application has on the network and on other applications, it is important to understand the application's traffic profile. To be meaningful a traffic profile needs to define at least the average rate and the burst characteristic of the application, over a defined time interval; this could be viewed as a virtual simple token bucket (as described in Chapter 2, Section 2.2.3). It can, however, easily be shown that different traffic profiles can share the same average rate and burst characteristics, hence more complex traffic profile descriptors are possible. Some applications will have a constant bit rate (CBR), which means that the burst characteristic of the traffic would relatively be the same over any time interval. Other applications may be described as variable bit rate (VBR), where the burst characteristic is relatively larger over smaller time intervals.

While there are clearly too many applications to describe them all, we consider the most common applications or application types, which impose the tightest SLA requirements on the network. In practice, most applications that have explicit SLA requirements will fall into one of the following categories, or will have SLA requirements, which are similar to one of those categories described:

- voice over IP
- video streaming
- video conferencing
- throughput-focussed TCP applications
- interactive data applications
- on-line gaming.

1.3.1 Voice over IP

Voice over IP (VoIP) is most commonly transported as a digitally encoded stream using the Real-time Protocol (RTP) [RFC3550] over UDP; RTP is the transport layer protocol, which deals with the delivery of the VoIP bearer stream from sender to receiver. Signaling protocols such as the Session Initiation Protocol (SIP) [RFC3261] may be used to set up the RTP bearer streams and to determine the media formats (i.e. codecs) that will be used. The key factors that determine the impact that variations in networks SLA characteristics such as delay and loss have on VoIP are the codec that is used to encode the signal and the specific details of the end-system implementation. For example, some codecs may be less tolerant to loss than others, while a poor end-system implementation may be less tolerant to jitter.

VoIP codecs convert analog voice signals into a digital bit stream at the sender and convert them back to an analog audio signal at the receiver. The most widely used codecs are those defined by the ITU G.71x and G72x standards. The simplest waveform-based codecs, such as that defined by ITU standard G.711, use pulse code modulation (PCM) where the analog signal is sampled at regular intervals; the samples are quantized into a set of discrete values to produce the encoded digital signal. More advanced codecs, such as that defined by ITU standard G.726, use Adaptive Differential PCM (ADPCM). ADPCM predicts the next sample from previous samples and then quantizes only the difference between the actual sample value and the prediction; consequently, ADPCM produces a lower bit rate signal than PCM at equivalent quality.

Frame-based codecs, such as ITU G.729 and ITU G.723, use more complicated techniques such as Algebraic Code Excited Linear Prediction (ACELP). ACELP breaks a sampled input signal into blocks of samples; these blocks or frames, which are typically 20 ms, are processed as whole units. In processing a frame, the encoder uses a technique called analysis-by-synthesis to determine which input parameters, when passed through a synthesizing filter, would result in reconstructed speech closest to the original speech signal. The encoder then uses a codebook to reference the inputs to the filter; the reference is sent to the decoder, which shares the same codebook, and which applies the respective inputs to the same synthesis filter to reconstruct the speech. There are a number of algorithms that have derived from ACELP, including Low-Delay Code Excited Linear Prediction (LD-CELP) and Conjugate Structure ACELP (CS-ACELP).

The codecs available for VoIP vary in complexity, in the bandwidth they need, and in the delivered call quality perceived by the end-user. Algorithms that are more complex may provide better perceived call quality, but may incur longer processing delays; Figure 1.5

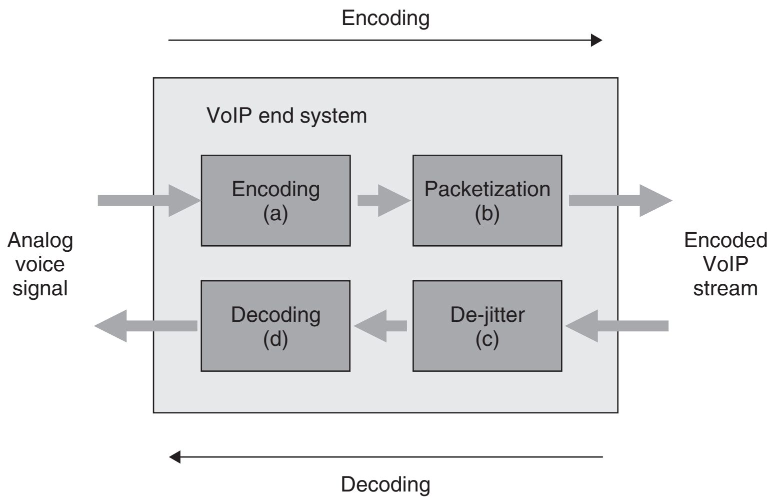

Figure 1.5 VoIP end-systems components of delay

shows the functional components in VoIP end-systems, which contribute to delay.

Some codecs use compression in order to reduce the bandwidth required for a VoIP call, which inevitably results in the loss of detail of the original signal, hence in general the better the call quality required, the more bandwidth that will be required per call. The table in Figure 1.6 compares characteristics of some of the more common VoIP codecs.

ITU-T Codec	Codec type	Maximum codec delay (ms) (a1 d)	Bitrate (bps)	Packetization interval (ms) (b)	pps	Payload size (bytes)	IP pkt size (bytes)[i]	IP bps
G.711	PCM	0.375	64 000	10	100	80	120	96 000
G.711	PCM	0.375	64 000	20	50	160	200	80 000
G.711	PCM	0.375	64 000	30	33.33	240	280	74 659
G.723.1	ACELP	97.5	5 300	30	33.33	20	60	15 998
G.723.1	ACELP	97.5	5 300	15	16.67	40	80	10 669
G.726.16	ADPCM	0.375	16 000	10	100	20	60	48 000
G.726.16	ADPCM	0.375	16 000	20	50	40	80	32 000
G.726.16	ADPCM	0.375	16 000	30	33.33	60	100	26 664
G.726.24	ADPCM	0.375	24 000	10	100	30	70	56 000
G.726.24	ADPCM	0.375	24 000	10	50	60	100	40 000
G.726.24	ADPCM	0.375	24 000	10	33.33	90	130	34 663
G.726.32	ADPCM	0.375	32 000	10	100	40	80	64 000
G.726.32	ADPCM	0.375	32 000	20	50	80	120	48 000
G.726.32	ADPCM	0.375	32 000	30	33.33	120	160	42 662
G.726.40	ADPCM	0.375	40 000	10	100	50	90	72 000
G.726.40	ADPCM	0.375	40 000	20	50	100	140	56 000
G.726.40	ADPCM	0.375	40 000	30	33.33	150	190	50 662
G.728	LD-CELP	1.875	16 000	10	100	20	60	48 000
G.728	LD-CELP	1.875	16 000	20	50	40	80	32 000
G.728	LD-CELP	1.875	16 000	30	33.33	60	100	26 664
G.729A	CS-ACELP	35	8 000	10	100	10	50	40 000
G.729A	CS-ACELP	35	8 000	20	50	20	60	24 000
G.729A	CS-ACELP	35	8 000	30	33.33	30	70	18 665

Figure 1.6 VoIP codec characteristics[2]

The impact that different SLA metric parameters have on VoIP applications is considered in the following sections.

1.3.1.1 VoIP: Impact of Delay

For VoIP the important delay metric is the one-way end-to-end (i.e. from mouth-to-ear) delay, in each direction. The main impact that end-to-end delay has on VoIP is to the interactivity of conversational speech. If the delay is too high, participants find it difficult to discern the difference between natural pauses in speech and the delays introduced by the system. If they mistake system delays for pauses in conversation and take these delays as their cue to begin to speak, by the time their words arrive at the other end, the other speaker may have already started to speak with the result that the normal protocol of conversation breaks down. Excessive end-to-end delay can also impair the effectiveness of mechanisms used for echo-cancellation.

The goal commonly used in designing networks to support voice over IP (VoIP) is the target specified by ITU-T recommendation G.114 [G.114], which uses the E-model (see Section 1.2.7.1) to estimate the effects of delay on mouth-to-ear speech transmission quality. Recommendation G.114 suggests that ~150 ms of end-to-end one-way delay is sufficient to ensure that users will be very satisfied for most applications of telephony. Higher delays may also be acceptable, but with a consequent reduction in user satisfaction, with delays exceeding 400 ms generally considered unacceptable, as shown in the table in Figure 1.7.

Network delay is only one component of the ear-to-mouth delay that impacts a VoIP call. Hence, having determined what the maximum acceptable ear-to-mouth delay is for a particular VoIP service, a network QOS design should take this budget and apportion it to the

Ear-to-mouth delay (D)	R factor	Objective MOS
$D < 150$ ms	80–89	5
150 ms $< D <$ 250 ms	70–79	4
250 ms $< D <$ 325 ms	60–69	3
325 ms $< D <$ 425 ms	50–59	2
$D >$ 425 ms	90–100	1

Figure 1.7 ITU G.114 Determination of the effects of absolute delay by the E-model

various components of network delay (propagation delay through the backbone, scheduling delay due to congestion, and the access link serialization delay) and end-system delay (due to VoIP codec and de-jitter buffer). The example timeline in Figure 1.8 shows the components of delay, which impact the ear-to-mouth delay of a VoIP service, using typical values for each component.

The codec delays are dependent upon the type of codec used. The table in Figure 1.6 lists the maximum theoretical one-way delay introduced by codec-related processing for a number of different codecs; in practice VoIP end-systems may incur an additional 5–20 ms of delay, depending upon the specific implementation. One-way network delays of 35–100 ms are typically targeted for high quality ("toll quality") VoIP services, in order to ensure that an ear-to-mouth delay of 150 ms can be achieved; an example VoIP delay budget is given in Chapter 3, Section 3.2.2.1.1 QOS mechanisms are typically employed to ensure these targets can be met. Higher delays may be tolerated for lower quality services.

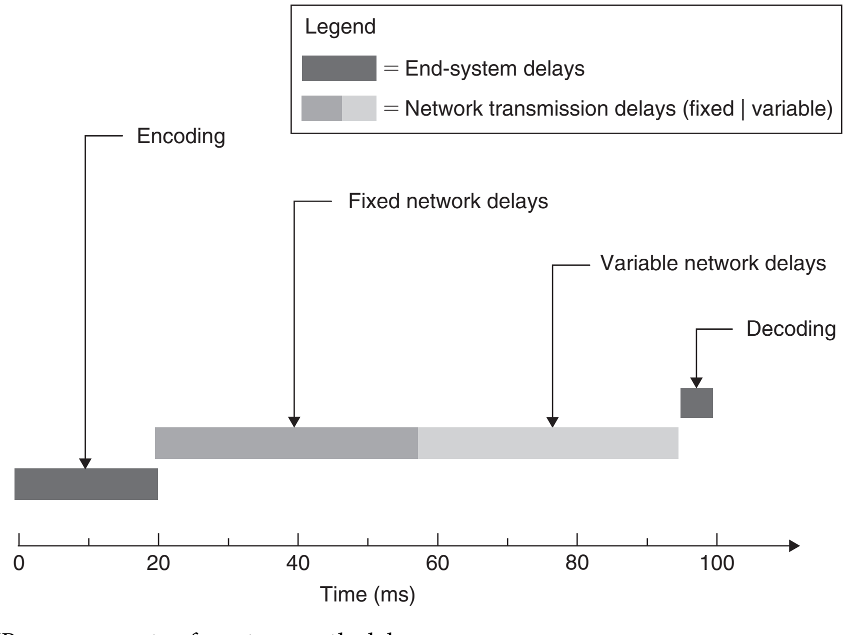

Figure 1.8 VoIP: components of ear-to-mouth delay

1.3.1.2 VoIP: Impact of Delay-jitter

It is a common misconception that jitter has a greater impact on the quality of VoIP calls than network delay. An understanding of how VoIP end-systems deal with delay and jitter is needed to understand why this is incorrect.

Applications which are susceptible to jitter, such as VoIP, use de-jitter buffers (also known as jitter buffers and play-out buffers) to compensate for jitter in packet arrival and for out-of-order packets. This is required because the decoding of the received signal is a synchronous process, and hence data must be fed into the decoder at a constant rate. De-jitter buffers remove delay variation by turning variable network delays into constant delays at the destination end-systems. If the de-jitter buffer play-out delay is set either arbitrarily large or arbitrarily small, then it may impose unnecessary constraints on the characteristics of the network or may affect the quality of the VoIP service. A play-out delay set too large adds unnecessarily to the end-to-end delay, as shown in Figure 1.9, meaning that less of the ear-to-mouth delay budget is available to be apportioned to the network and hence that the network needs to support a tighter delay target than practically necessary.

If the de-jitter buffer play-out delay is too small to accommodate the network jitter then buffer underflows can occur; an underflow is

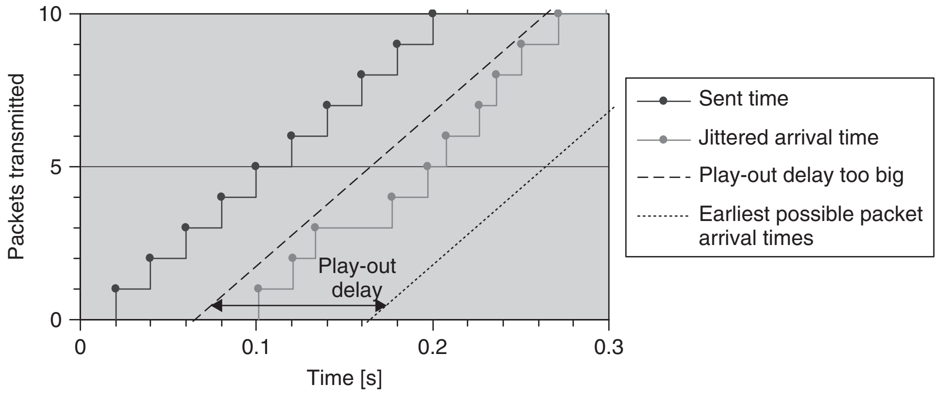

Figure 1.9 VoIP play-out delay unnecessarily large

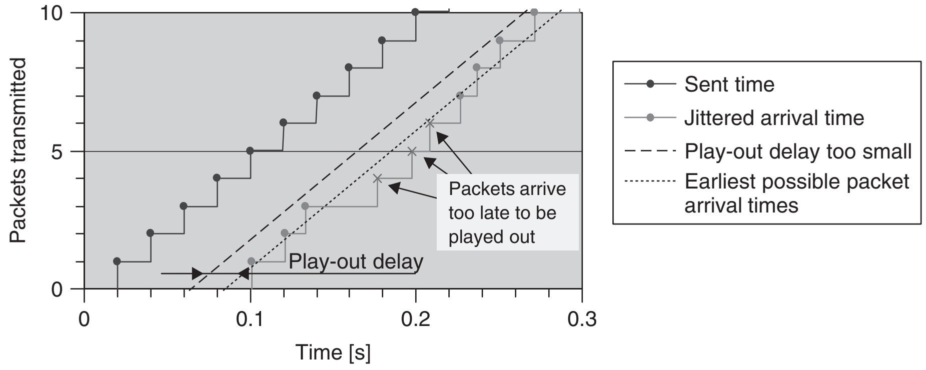

Figure 1.10 VoIP play-out delay too small

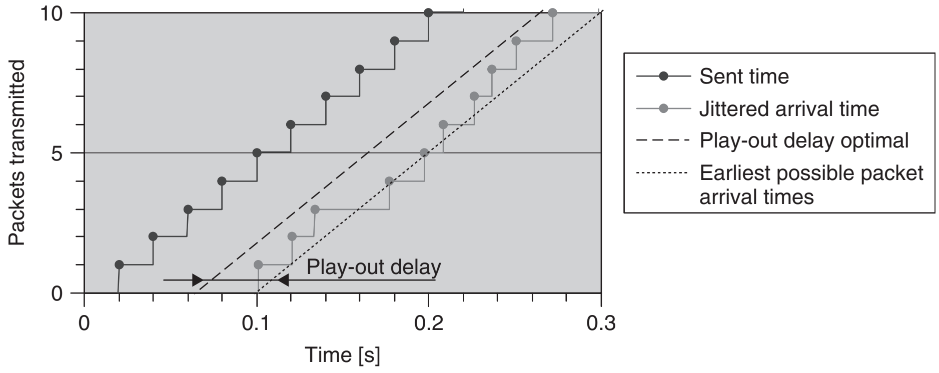

Figure 1.11 Optimal VoIP play-out delay

where the buffer is empty when the codec needs to play out a sample, and is effectively realized as a "lost" packet as shown in Figure 1.10.

Most VoIP end-systems use adaptive de-jitter buffers, which aim to overcome these issues by dynamically tuning the play-out delay to the lowest acceptable value, as shown in Figure 1.11.

Well-designed adaptive de-jitter buffer algorithms should not impose any unnecessary constraints on the network design if they display the following characteristics:

- increasing the play-out delay to the current measured jitter value following an underflow, and using packet loss concealment (see

Section 1.3.1.3) to interpolate for the "lost" packet and for the play-out delay size increase

- if the play-out delay can decrease then it should do so slowly when the measured jitter is less that the current buffer play-out delay.

Where such adaptive de-jitter buffers are used, they dynamically adjust to the maximum value of network jitter. In this case, the jitter buffer does not add delay in addition to the worst-case end-to-end network delay.

1.3.1.3 VoIP: Impact of Loss

Packet Loss Concealment (PLC) is a technique used to mask the effects of lost or discarded VoIP packets; an understanding of PLC is needed to understand the impact that packet loss has on the resultant quality of a VoIP call. The method of packet loss concealment used depends upon the type of codec used.

A simple method of packet loss concealment, used by waveform codecs like G.711 (PLC for G.711 is defined in G.711 Appendix I), is to replay the previously received sample; the concept underlying this approach is that, except for rapidly changing sections, the speech signal is locally stationary. This technique can be effective at concealing the loss of up to approximately 20 ms of samples. The packetization interval determines the size of samples contained within a single packet; assuming a 20 ms packetization interval, the loss of two or more consecutive packets will result in a noticeable degradation of voice quality.[3] From a network design perspective, it is important to note that a design decision to use a 30 ms packetization interval, for a given probability of packet loss, could result in worse perceived call quality than a 20 ms packetization interval, as with a 30 ms interval PLC may not be able to effectively conceal the loss of a single packet. Hence, there is a network design trade-off to be considered; larger packetization intervals may reduce the bandwidth of a VoIP call (there is less IP overhead due to more samples being carried in a single packet) but may also result in lower call quality for a given loss rate.

Low bit rate frame-based codecs, such as G.729 and G.723, use more sophisticated PLC techniques, which can conceal up to 30–40 ms of loss with "tolerable" quality, when the available history used for the interpolation is still relevant. Concealment becomes problematic with short phonemes – *the smallest phonetic unit in a language* – where 30 ms of samples can be over half of a phoneme and previous and subsequent samples may not provide enough information about the lost sample to allow it to be effectively concealed. With frame-based codecs, the packetization interval will determine the number of frames carried in a single packet. Similarly as for waveform-based codecs, if the packetization interval were greater than the loss that the PLC algorithm can interpolate for, then PLC would not be able to conceal the loss of a single packet effectively.

Hence, to summarize the impact that packet loss has on VoIP, with an appropriately selected packetization interval (20–30 ms depending upon the type of codec used) a loss period of one packet may be concealed but a loss period of two or more consecutive packets may result in a noticeable degradation of voice quality. The loss distance targeted for a particular service is a choice for the service provider.

In supporting VoIP services, it is essential to understand what impact these targets have on the network design in practice; consider the impact of the possible causes of packet loss previously defined in Section 1.2.3:

- *Congestion*. When congestion that impacts VoIP traffic occurs, queues build up and VoIP packets are dropped. If congestion occurs, it is not practically possible to engineer the network to ensure that consecutive packets from a single VoIP call are not dropped, nor to ensure that if they are dropped that it is with a controlled distribution. For this reason, networks supporting VoIP are designed to be congestionless from the perspective of the VoIP traffic; that is the available capacity for VoIP traffic is able to cope with the peak of the offered VoIP traffic load. QOS mechanisms, admission control techniques and appropriate capacity planning techniques are deployed to ensure that no packets are lost due to congestion.

- *Lower layer errors*. As described in Section 1.2.3, physical layer bit errors may cause packets to be dropped due to link layer or transport-layer checksums. Hence, bit errors will generally result in packet loss, meaning each packet will either arrive correctly or not at all. QOS mechanisms cannot have any impact on loss due to lower level errors; hence, where the underlying network transport infrastructure cannot meet the loss distance targets required for the VoIP service, PLC techniques will be required.

  Consider for example a typical BER offered for a leased line service of $1 * 10^{-9}$ and assume a random error distribution, that each error causes a lost packet, and that a G.711–20 ms codec is used which produces 200-byte packets at 50 pps; the resultant PLR would be $1 * 10^{-9} * 200 * 8 = 1.6 * 10^{-6}$. Without PLC this would result in an effective loss distance of $1/(1.6 * 10^{-6} * 50\text{pps} * 60)$ seconds = $\sim$208 minutes, which is better than typical service targets, which are in the order of 1 audible artifact every 30 minutes. PLC would further increase the attained loss distance.

  As an alternative example, consider a typical BER offered for an ADSL service of $1 * 10^{-7}$; this would equate to a PLR of $1 * 10^{-7} * 200 * 8 = \sim 1.6 * 10^{-4}$ which, without PLC, would result in a loss distance of ~2 minutes, which is an order of magnitude less than typical service targets. With PLC interpolating for the loss of a single packet, the probability of an audible impairment (due to two consecutive packets lost) would be $(1.6 * 10^{-4})^2 = \sim 2.6 * 10^{-8}$, i.e. an effective loss distance of $1/(2.6 * 10^{-8} * 50\text{pps} * 60 \text{ seconds} * 60 \text{ minutes} * 24 \text{ hours}) = \sim 9$ days! Hence, even for ADSL services, the impact on VoIP services of bit errors is unlikely to be significant.

- *Network element failures*. Network element failures may cause packets to be dropped until connectivity is restored around the failed network element. The resulting loss period depends upon the underlying network technologies that are used, as discussed in Section 1.2.3.

  In a "plain" IP (i.e. non-MPLS) deployment, even in a well-designed network where the IGP convergence time is 100s of milliseconds, the packet loss following a network element failure is too significant to be concealed and an audible glitch will result.

Where MPLS TE FRR or equivalent techniques are deployed, reducing the loss of connectivity following network element failures to within 50 ms, PLC may be able to interpolate for the resultant packet loss, but this is not guaranteed.

- *Loss in the application end-systems*. Loss due to buffer underflows and overflows can be prevented through careful design both of the network and the application end-systems, as discussed in Section 1.3.1.2.

Therefore, in practice, networks supporting VoIP should typically be designed for very close to zero percent VoIP packet loss. QOS mechanisms, admission control techniques and appropriate capacity planning techniques are deployed to ensure that no packets are lost due to congestion with the only actual packet loss being due to layer 1 bit errors or network element failures. Where packet loss occurs, the impact of the loss should be reduced to acceptable levels using PLC techniques.

1.3.1.4 VoIP: Impact of Throughput

VoIP codecs generally produce a constant bit rate stream, with bandwidths as shown in Figure 1.6; that is, unless silence suppression is used. Silence suppression, which is also known as voice activation detection (VAD), prevents the transmission of packets carrying "silent" samples. Silence suppression becomes active when it detects periods of silence from the microphone that exceed defined thresholds; when silence suppression is active it prevents the encoder output from being sent to the far end. When silence suppression is active for a leg of a VoIP call, the bandwidth used for that leg of the call is almost zero. As most conversational speech contains approximately 50% silence, this can significantly reduce the average bandwidth used for a call; however, the peak bandwidth used for the call remains unchanged.

As discussed in Section 1.3.1.3, networks supporting VoIP should typically be designed for very close to zero percent VoIP packet loss, and hence are designed to be congestionless from the perspective of the VoIP traffic. This means that the available capacity for VoIP traffic must be able to cope with the peak of the offered VoIP traffic load. Further, in order to ensure that the service availability is maintained, this peak load must be able to be supported without loss while maintaining the required delay and jitter bounds for the VoIP traffic. The bandwidth provisioning required to achieve this is discussed in more detail in Chapter 3, Section 3.2.2.8 and Chapter 6, Section 6.1.3.

Even if VoIP capacity is provisioned to support the peak load, the VoIP service may be statistically oversubscribed. For example, assuming that a link could support a maximum of 30 concurrent VoIP calls while ensuring that the delay, jitter and loss targets are still met, as only a portion of the total number of end-users will have an active call at any particular time, many more than 30 end-users may be supported, resulting in bandwidth efficiencies from statistical multiplexing. If more than 30 users were supported and at peak times, the VoIP load may exceed the available VoIP capacity, then the service to all calls in progress may be degraded. If the probability of this occurring is high enough, then an admission control system may be needed to limit the number of VoIP calls that can be concurrently set up such that the available capacity is never exceeded in practice; admission control is discussed in more detail in Chapter 4.

It is further noted that lower bit rate codecs typically incur greater codec delays, hence when opting for lower bit rate codecs to save bandwidth it should be understood that there may be a consequent increase in terms of ear-to-mouth delay.

1.3.1.5 VoIP: Impact of Packet Re-ordering

VoIP traffic is not commonly impacted by packet re-ordering, as the magnitude of re-ordering would need to be very significant to affect a VoIP flow whose inter-packet gap is a multiple of 20 ms, for example. It is, however, noted that in addition to the impact that it has on application throughput, per-packet load balancing, which is a common cause of packet re-ordering, can also increase the jitter that is experienced within a flow due to the different delays of alternate paths; this effect can impact VoIP services.

1.3.2 Video

1.3.2.1 Video Streaming

With video streaming applications, a client requests to receive a video that is stored on a server; the server streams the video to the client, which starts to play out the video before all of the video stream data has been received. Video streaming is used both for "broadcasting" video channels, which is often delivered as IP multicast, and for video on demand (VOD), which is delivered as IP unicast.

IP-based streaming video is most commonly transported as a data stream encoded using standards defined by the Motion Picture Expert Group (MPEG) and transported using RTP over UDP. MPEG defines the encoding used for the actual video stream, while [RFC2250, RFC 2343, and RFC3640] define how real-time audio and video data are formatted for RTP transport. RTP is the transport layer protocol, which deals with the delivery of that stream from sender to receiver. Protocols such as the Real-time Streaming Protocol (RTSP) [RFC2326] may be used to set up the RTP streams. While other encodings and transport protocols may be used, as this case is the most widespread we consider it in more detail and note that the principles discussed are generally applicable to other encoding schemes and transport layer protocols also.

The MPEG committee [MPEG] is a working group of the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) working on the development of standards for digital audio and video. MPEG have been responsible for producing a number of standards that can be used for IP-based services including MPEG-2 [MPEG-2] (the video part of which is the same as ITU-T standard H.262), which is used for broadcast quality video encoding including digital television services, and the newer MPEG-4 Advanced Video Coding (AVC) [MPEG-4] (which is also known as MPEG-4 Part 10, and which is technically identical to ITU-T standard H.264) standard which was designed for Internet audio and video encoding. MPEG-2 encoding is most widely used today for television applications; however, newer encoding schemes such as MPEG-4 and the Society of Motion Picture Television Engineers (SMPTE) VC-1

[VC-1] (which was previously known as VC-9 and is the standardized version of Windows Media™ 9) are likely to become more widespread as they offer a potential bit rate reduction by two times over MPEG-2, for comparable quality.

An MPEG encoder converts and compresses a video signal into a series of pictures or frames; as there is generally only a small amount of change between one frame and the next it is possible to compress the video signal significantly by transmitting only the differences. There are three different types of MPEG frames that are used:

- *"I"-frames*. Intra or "I"-frames carry a complete video frame and are coded without reference to other frames. An I-frame may use spatial compression; spatial compression makes use of the fact that pixels within a single frame are related to their neighbors. Therefore, by removing spatial redundancy, the size of the encoded frame can be reduced and prediction can be used in the decoder to reconstruct the frame. A received I-frame provides the reference point for decoding a received MPEG stream.
- *"P"-frames*. Predictive coded or "P"-frames are coded using motion compensation (temporal compression) by predicting the frame to be coded from a previous "reference" I-frame or P-frame. P-frames can provide increased compression compared to I-frames with a P-frame typically 10–30% the size of an associated I-frame.
- *"B"-frames*. Bidirectional or "B"-frames use the previous and next I- or B-frames as their reference points for motion compensation. B-frames provide further compression, still with a B-frame typically 5–15% the size of an associated I-frame.

Frames are arranged into a Group of Pictures or GOP; for example, the European PAL (Phase-Alternating Line) MPEG-2 video format uses a GOP size of 15, while the North American NTSC (National Television Systems Committee) format uses a GOP size of 18. As the frame rate for PAL is 25 frames per second (fps) and for NTSC is 29.97 fps, each GOP will typically encode (15/25) = ~(18/30) = ~0.6 seconds of video. There are many possible GOP structures and the makeup of I, P and B frames within the GOP is determined by the format of the source video signal, any bandwidth constraints on the encoded video stream (which determines the required compression ratio), and possibly constraints on the encoding/decoding delay. Each GOP has one I-frame, and typically 2-to-14 P-frames and 2-to-10 B-frames. A regular GOP structure can be described by the number of frames in the GOP (the GOP size) and the spacing of P-frames within the GOP. A typical GOP structure with GOP size of 15 frames and P-frame spacing of 3 (denoted as a 15/3 GOP structure) is shown below:

$$B_1\ B_2\ I_3\ B_4\ B_5\ P_6\ B_7\ B_8\ P_9\ B_{10}\ B_{11}\ P_{12}\ B_{13}\ B_{14}\ P_{15}$$

The GOP structure shown above is in the order of display; to allow for backward prediction, the encoder re-orders the frames from display order so that B-frames are transmitted after the previous and next frames it references. The resulting order of frames sent to the decoder is:

$$I_3\ B_1\ B_2\ P_6\ B_4\ B_5\ P_9\ B_7\ B_8\ P_{12}\ B_{10}\ B_{11}\ P_{15}\ B_{13}\ B_{14}$$

This re-ordering introduces a delay both on encoding and on decoding dependent on the number of consecutive B-frames.

Unlike with VoIP where codec implementations are very specifically defined, with streaming video there is significant scope for variation in the specific way that an MPEG stream may be encoded, even for a single type of encoding. The specific GOP structure used to encode a video stream can have a major impact on the effect that network loss, latency and throughput have on the video reproduction at the receiver.

1.3.2.1.1 Video Streaming: Impact of Delay

For video streaming, the important delay metric is the one-way end-to-end delay from streaming server to client. The main constraint that end-to-end network delay and jitter have on streaming video is on end-user "interactivity," or the "finger-to-eye" delay. To understand this better we need to consider some of the different types of video streaming applications separately:

1.3.2.1.1.1 Broadcast Video Services Broadcast television services delivered over IP (also known as IPTV) commonly use IP multicast. For IPTV services, the impact that end-to-end delay has is on the time it takes for the end-user to change from one TV channel to another, which is referred to as the "channel change time" or "channel zapping time." Typically, channel change times of 1–2 seconds are targeted (see Section 1.3.3.2 for requirements for interactive applications) although visual feedback is typically provided to the user indicating that the command is being processed within a few hundred milliseconds.

Assuming a broadcast video service being delivered using IP multicast to a receiver – which could be a set-top box (STB) for example – where each channel is a separate multicast group, the overall channel change time is made up of a number of components:

1. *Remote control and STB processing*. The remote control sends the channel change signal to the STB; typically, this takes a few milliseconds. The STB receives the channel change signal, processes the command and issues IGMP group leave and join requests to the network. This delay will be dependent upon the particular STB implementation, but would typically take a few tens of milliseconds; for this example, we assume a worst case of 50 ms.
2. *Network transmission delay*. This is the network transmission delay for the IGMP messages from the STB to the first multicast aware device, which needs to process the IGMP messages; this includes delays due to serialization, switching, propagation and queuing. As the STB and closest multicast aware device are usually physically located relatively near to each other, and QOS mechanism are employed to control queuing delays and ensure IGMP messages are not dropped, this network transmission delay is typically sub-100 ms.
3. *Multicast processing*. When the first multicast aware device receives the IGMP group leave, assuming fast leave processing, it stops

sending the multicast stream from the previous channel on the respective port. When the IGMP group join is received, assuming the multicast stream for the requested channel traffic already exists at the router, the traffic is copied to the egress port. If this assumption is not correct, additional latency may be incurred due to multicast signaling, while the multicast stream is populated to the first multicast aware device.

Multicast processing is dependent upon the particular implementation in the multicast device, but typically takes a few tens of milliseconds in a good implementation; for this example, we assume a worst case of 100 ms.

4. *Network transmission delay*. This is the network transmission delay for the multicast traffic stream to reach the STB; this includes delays due to serialization, switching, propagation and queuing. As the STB and closest multicast aware device are usually physically located relatively near to each other, and QOS mechanism are employed to control queuing delays and ensure that multicast video traffic is not dropped, this delay is typically sub-100 ms.

5. *STB Buffering/processing*. The STB needs to buffer the received video stream, and perform a number of functions before the video can be played out; depending upon the specific STB implementation, some of these functions may be performed in parallel:
   - *De-jitter buffer*. As for VoIP, de-jitter buffers are used in IP-based digital video systems in order to turn variable network delays into constant delays at the receiver. If the de-jitter buffer play-out delay is set either arbitrarily large or arbitrarily small, then it may impose unnecessary constraints on the characteristics of the network or may impact the quality of the video service. The delay incurred by the de-jitter buffer in addition to the worst-case end-to-end network delay depends upon the maximum network jitter and the STB play-out buffer sizing. Considerations on jitter for IP-based video applications are discussed in more detail in Section 1.3.2.1.2. We assume a worst-case play-out delay of 100 ms.

- *FEC or real-time retransmission delay*. If the video system supports FEC or real-time retransmission to protect against network packet loss, then this will incur a delay. This delay will typically be 100–200 ms; see Section 1.3.2.1.3.
- *Decryption delay*. For encrypted streams, the per-channel decryption keys are delivered to each STB periodically as conditional access table information in the transport stream packets; these keys must be received before the channel can be decoded. The frequency of the key distribution therefore has an impact on the channel change time; this delay is typically 200 ms.
- *MPEG decoder buffer*. With MPEG, the amount of data required to represent an image depends upon the image complexity and therefore the output rate of the decoder buffer is not constant. Hence, a buffer is required to ensure that the decoder does not underflow. This buffer is typically 500–1000 ms.
- *IBB frame delay*. The decoder needs to wait until it has received an IBB frame sequence before it can start decoding. For a 15/3 GOP structure, typical for MPEG-2, at 25 fps the worst-case delay for an IBB frame sequence would be $(15 + 2)/25 = \sim 680$ ms. Longer GOP structures may improve the compression achievable but will also increase the worst-case delay for an IBB-frame sequence, and hence have an impact on the channel change time.

For this example, we assume a total delay incurred at the STB of 1000 ms.

The timeline in Figure 1.12 illustrates the components of channel change time delay described above.

1.3.2.1.1.2 Video-on-demand Services Video-on-demand (VOD) and network personal video recorder (PVR) services are commonly delivered as unicast. For VOD services the end-to-end delay impacts the finger-to-eye delay, i.e. the response time it takes for user requests to be translated into actions visible to the end-user; for example, how long it takes after pressing play for a VOD to start. Typically,

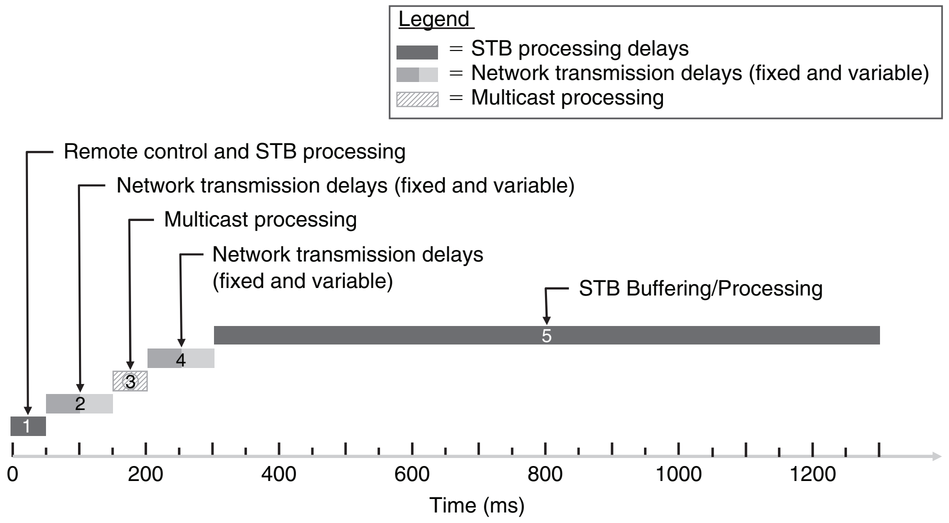

Figure 1.12 Broadcast video channel change time delay components

response times of approximately 1 second are targeted (see Section 1.3.3.2 for requirements for interactive applications), although visual feedback is typically provided to the user indicating that the command is being processed within a few hundred milliseconds.

Assuming a video-on-demand service being delivered over IP unicast to a receiver, which could be a set-top box (STB) for example, the overall response time is made up of a number of components:

1. *Remote control and STB processing*. The remote control sends the control signal to the STB; typically, this takes a few milliseconds. The STB receives the signal, processes the command, and issues the appropriate control request to the video middleware, which is responsible for managing the presentation and delivery of VOD streams. This delay will be dependent upon the particular STB and middleware implementations but would typically take a few tens of milliseconds; we assume a worst case of 50 ms in this example.

2. *Network transmission delay*. This is the network transmission delay for the control messages from the STB to the video middleware; this includes delays due to serialization, switching, propagation and queuing. QOS mechanisms are employed to regulate queuing delays and ensure control messages are not dropped, and hence this network transmission delay is typically sub-100 ms.

3. *Middleware processing*. This is the delay for the video middleware application to process the received request and instruct the VOD server to start streaming the requested VOD. The middleware processing delay is dependent upon the particular middleware implementation, but typically takes a few hundred milliseconds.

4. *Network transmission delay*. This is the network transmission delay for the unicast VOD stream to reach the STB; this includes delays due to serialization, switching, propagation and queuing. QOS mechanisms are employed to regulate queuing delays and ensure control messages are not dropped, and hence this network transmission delay is typically sub-100 ms.

5. *STB buffering/processing*. The STB needs to buffer the received video stream, and perform a number of functions before the video can be played out; depending upon the specific STB implementation, some of these functions may be performed in parallel:
   - de-jitter buffer
   - FEC or real-time retransmission delay
   - decryption delay
   - MPEG decoder buffer.

   The above functions are the same as described for the broadcast video example described in Section 1.3.2.1.1.1. It is noted, however, that for VOD services, unlike broadcast video applications where the STB may need to wait for an I-frame before a stream can start to be decoded, an IBB frame sequence is the first frame that is sent in a VOD stream. Hence the STB buffering and processing delay can be significantly less in the case of VOD, than the channel change time in the broadcast video case. For this example, we assume a total delay incurred at the STB of 500 ms.

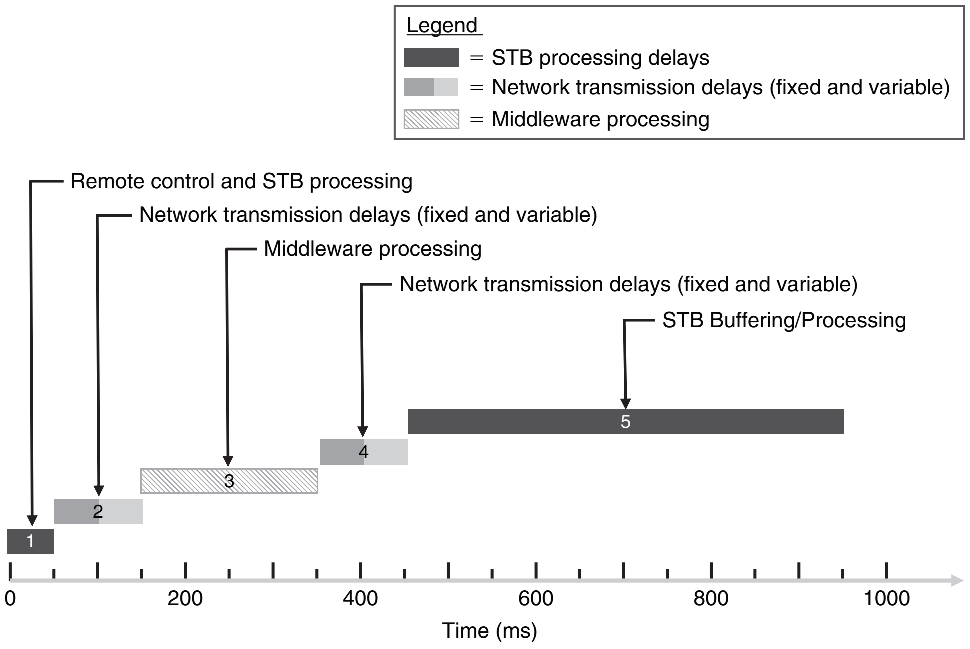

Figure 1.13 VOD response time delay components

The timeline in Figure 1.13 illustrates the components of the VOD response time described above.

Hence, for video streaming applications, one-way network delays of 100 ms are typically targeted in order to try to achieve overall channel change times or VOD response times of 1–2 seconds. QOS mechanisms are typically employed to ensure these targets can be met.

1.3.2.1.2 Video Streaming: Impact of Delay-jitter

Digital video decoders used in streaming video receivers need to receive a synchronous stream, typically with jitter tolerances of only ±500 ns, in order to decode without visible impairments. Such jitter tolerances are not achievable natively in IP networks, hence as for VoIP, broadcast video services use de-jitter buffers (also known as play-out buffers) in receivers to remove delay variation caused by the network and turn variable network delays into constant delays such that the tolerances required by the decoder can be met.

To guarantee that no packet drops are caused by network jitter, QOS mechanism are employed to control queuing delays and hence bound jitter; an STB play-out buffer should be sized at least equal to the resulting maximum possible value of network jitter. As for VoIP, if the video play-out buffer was appropriately sized to the maximum value of network jitter, jitter would not add delay on play-out in additional to the worst-case end-to-end network delay. For streaming video applications, however, unlike for VoIP, the size of jitter buffers is often statically defined. Hence, if the play-out buffer is too small to accommodate the maximum network jitter then buffer underflows can occur; an underflow is where the buffer is empty when the decoder needs to process a frame, and is effectively realized as a lost packet, which may cause a video impairment. A play-out buffer set too large adds unnecessarily to the end-to-end delay, which may increase the channel change time or decrease VOD responsiveness.

Hence, a holistic engineering approach is needed in order to optimize channel change time or VOD responsiveness, employing QOS mechanisms to control queuing delays and hence bound the maximum network jitter, while ensuring that the STB play-out buffer is not set to significantly greater than this value. An STB play-out buffer that is more than 100 ms greater than the maximum network jitter can be considered excessive.

1.3.2.1.3 Video Streaming: Impact of Loss

Each MPEG frame is transported as a number of MPEG transport steam (MPEG-TS) packets, which are typically 188 bytes long. Each IP packet typically carries between 1 and 7 MPEG-TS packets, and an MPEG encoded frame will span multiple IP packets, hence, without employing techniques for loss concealment, the loss of a single IP packet will result in the loss of a complete MPEG frame. Without employing loss concealment techniques, the loss of a frame will generally result in a visible impairment, although the nature of the impairment will depend upon the type of frame lost and on the GOP structure used. The loss of an I-frame can result in a visual impairment until the next I-frame is successfully received; the loss of a P-frame may impact several frames, while the loss of a B-frame may impact that frame only.

Therefore, there is a trade-off to be considered when deciding on which GOP structure to use to support a streaming video service; larger GOP size will give greater compression, which results in more or higher quality video content being transmitted for a given rate than a smaller GOP size; however, the visual impact of packet loss may be more significant for larger GOP, and there may be an impact on the interactivity, e.g. finger-to-eye delay.

The loss distance targeted for a particular video service is a choice for the service provider; however, targets of no more than one visible impairment per hour are typical, such as defined by the Digital Video Broadcasting Project (DVB) in draft [ETSI Ts 102 034], which specifies network requirements for the "Transport of DVB Services over IP." This bound on the rate of visible impairments directly translates into a bound on packet loss. Assuming a 3.7 Mbps MPEG stream with 1356-byte packets (based upon an MPEG transport stream carrying 7×188 bytes of video in each packet, plus 40 bytes for RTP, UDP and IP headers), gives a stream rate of ~350 pps. For this stream, a loss period of one hour would result in a required PLR of no more than $1/(350\,\text{pps} * 60\text{ seconds} * 60\text{ minutes}) = \sim 1 * 10^{-6}$. We consider what effect targets such as this have on the network design in practice by considering the impact on the possible causes of packet loss previously defined in Section 1.2.3:

- *Congestion*. For the same reasons as discussed in Section 3.1.3 for VoIP, networks supporting streaming video are designed to be congestionless from the perspective of the streaming video traffic; that is, the available capacity for streaming video traffic is able to cope with the peak of the offered video traffic load. QOS mechanisms, admission control and appropriate capacity planning techniques are deployed to ensure that no packets are lost due to congestion.
- *Lower layer errors*. As described in Section 1.2.3, physical layer bit errors may cause packets to be dropped due to link layer or transport-layer checksums. Where this is the case, bit errors will

generally result in packet loss, meaning each packet will either arrive correctly or not at all. Consider for example a typical BER offered for a SONET/SDH service of $1 * 10^{-12}$, and assume a random error distribution, and that each error causes a lost packet; for the video stream described above, this would result in a PLR of $1 * 10^{-12} * 1356 * 8 = \sim 1 * 10^{-8}$; this would result in a loss distance of over 79 hours, which is several orders of magnitude greater than the target. As an alternative example, consider a typical BER offered for an ADSL service of $1 * 10^{-7}$; this would equate to a PLR of $1 * 10^{-7} * 1356 * 8 = \sim 1 * 10^{-3}$, which would result in a loss distance of less than 3 seconds, and which is several orders of magnitude less than the target.

QOS mechanisms cannot have any impact on loss due to lower level errors, hence where the underlying network transport infrastructure cannot meet the loss distance targets required for the video service, loss concealment techniques may be required. There are two main techniques for loss concealment for streaming video:

- *Forward error correction (FEC)*. FEC relies on redundancy being built into the transmitted content stream in order to be able to reconstruct lost packets without the need for retransmission. The Professional-MPEG Forum's [PRO-MPEG] published code of practice (COP) number 3 recommends a scheme based on the approach defined in RFC 2733 [RFC2733] which specifies a FEC mechanism for protecting a RTP stream against lost RTP packets. In the Pro-MPEG Forum scheme, XOR operations are performed on a block of packets arranged in a matrix of D rows by L columns to generate redundant parity packets. At the receiver, the FEC information is used to recover from losses within a FEC block.

  The structure of the matrix that forms the FEC block impacts the loss burst size the FEC can protect against, the bandwidth overhead associated with the FEC stream and the delay caused by the FEC processing operation. Hence, it is important that the FEC parameters be configured to match the requirements of a particular service, taking into account the characteristics of the underlying network.

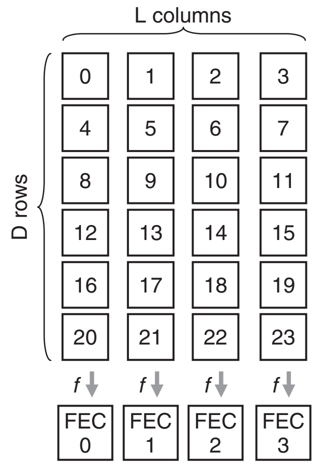

Figure 1.14 Professional-MPEG Forum's COP 3 1D FEC

The Pro-MPEG Forum scheme allows for both one-dimensional (1D) FEC, as shown in Figure 1.14, and two-dimensional (2D) FEC, as shown in Figure 1.15. The FEC information is carried in a separate stream (for 1D FEC) or streams (for 2D FEC) to the video bearer.

One-dimensional FEC is able to recover from 1 error within the FEC block and has an overhead of L/(L * D). The two-dimensional scheme is able to recover from a burst of L errors within a FEC block and has an overhead of (L + D)/(L * D). The additional delay incurred by either scheme is the time to transmit L * D packets. Larger D reduces the overhead at the cost of increasing the additional delay.

For example, assuming a 3.7 Mbps MPEG-2 stream consisting of ~350 pps each with a 1356-byte payload:

– Using a one-dimensional FEC with L = 4 and D = 6 provides the ability to recover from 1 error in 24 packets, while incurring 4/(4 * 6) = ~17% overhead and 24/330 = ~72 ms of additional delay due to the FEC processing operation.[4]

  Assuming a PLR of $1 * 10^{-3}$ (the previously calculated typical PLR for ADSL), with a random packet loss distribution, the

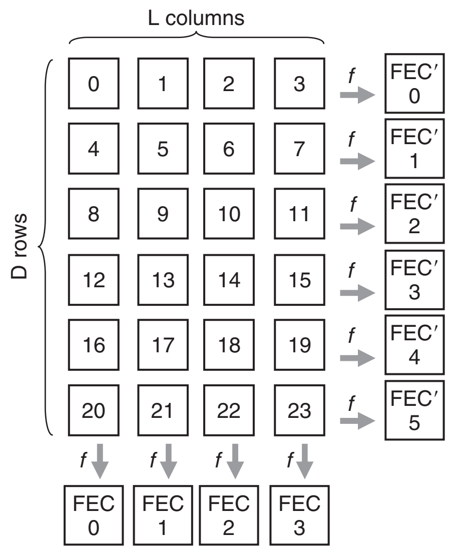

Figure 1.15 Professional-MPEG Forum's COP 3 2D FEC

probability of an unrecoverable loss (i.e. 2 lost packets) within a 24 packet block is $(1 * 10^{-3} * 24) * (1 * 10^{-3} * 23) = \sim 6 * 10^{-4}$; this results in a loss distance of ~5 seconds, which is still several orders of magnitude worse than the target.

– By comparison, using a two-dimensional FEC with L = 4 and D = 6 provides the ability to recover from a burst of 4 errors in 24 packets, while incurring (4 + 6)/(4 * 6) = ~42% overhead and 24/330 = ~72 ms of additional delay due to the FEC processing operation.

Assuming a PLR of $1 * 10^{-3}$, the probability of an unrecoverable loss (i.e. 5 lost packets) within a 24 packet block is $(1 * 10^{-3} * 24) * (1 * 10^{-} * 23) * (1 * 10^{-3} * 22) * (1 * 10^{-3} * 21) * (1 * 10^{-3} * 20) = \sim 5 * 10^{-9}$; this results in a loss distance of greater than ~155 hours, which is several orders of magnitude better than the target.

○ *Real-time retransmission*. Media streams that use RTP are to some extent resilient, in that receivers may use the mechanisms defined for the RTP control protocol (also known as RTCP) to report packet reception statistics and thus allow a sender to adapt its transmission behavior. Additional techniques are being defined within the IETF which extend the basic capabilities of RTCP to allow for faster feedback of packet loss from receivers to senders [RFC4584], such that lost packets may be retransmitted [RFC4588]. Within a defined time window, receivers can detect sequence number gaps in the received stream indicating lost packets and report these using RTCP negative acknowledgments (NACKs) to the sender, which retransmits the lost packets.

Real-time retransmission is a reactive scheme, which resends only those packets that are lost, hence it incurs minimal bandwidth overhead. A disadvantage of this approach, however, is that it adds delay equal to the RTT between the receiver and the retransmission source, to the potential worst-case delay that a packet may otherwise experience. Hence, real-time retransmission is only viable in cases where the RTT between the receiver and the retransmission source can be kept small, in order to avoid increasing the channel change time or decreasing VOD responsiveness.

It is noted, however, that some video end-systems disable the UDP checksum, or use "UDP-lite" [RFC3828], such that packets with data bit errors will be received including the error bits, on the premise that it can be better to receive a packet with a bit error, than to receive no packet at all [SINGH]. Currently, however, there is insufficient data on the performance of such implementations to quantify what impact this has on the received video service in practice.

- *Network element failures*. Network element failures may cause packets to be dropped until connectivity is restored around the failed network element; as discussed in Section 1.2.3 the resulting loss period depends upon the underlying network technologies that are used.

  In a "plain" IP (i.e. non-MPLS) deployment, even in a well-designed network where the IGP convergence time is subsecond, the packet loss following a network element failure is too significant to be concealed using any of the techniques described above. For example, assuming a 500 ms loss of connectivity and an impacted 3.7 Mbps MPEG-2 video stream at 350 pps, ~175 packets would be lost due to the outage. Even where MPLS TE FRR or equivalent techniques are deployed, reducing the loss of connectivity following network element failures to within 50 ms, the packet loss may be too significant to be viably concealed using any of the techniques previously described. For example, assuming a 50 ms loss of connectivity and an impacted 3.7 Mbps MPEG-2 video stream at 350 pps, ~18 packets would be lost due to the outage.

Where the impact of network outages is such that the loss distance targets required for the video service cannot be met, techniques using stream redundancy may provide a possible solution; available techniques are either based upon spatial or temporal redundancy:

- *Spatial redundancy*. Techniques using spatial (physical) redundancy send two streams between the sender and receiver over diverse network paths; diverse path routing can be assured using techniques such as MPLS traffic engineering. In normal working case network conditions, the receiver will effectively receive two copies of each packet, from which one will be selected. If a network failure impacts one of the transmitted streams, the receiver will continue to receive the other and the play-out of the video stream at the receiver will be uninterrupted.

  Schemes using physically redundancy incur a 100% bandwidth overhead per stream, and also must incur a delay on the received stream at least equal to the greatest difference in transmission delays between the two paths, which is likely to be negligible in practice.
- *Temporal redundancy*. Techniques using temporal redundancy break the stream to be transmitted into blocks: each block is then sent twice, separated in time. The block repetition pattern is such that within a time window greater than the block separation period the receiver should effectively receive two copies of each packet, from which one will be selected. If a network failure occurs, which causes a resulting loss of connectivity of less than the block separation period, at least one packet should be received and the play-out of the video stream at the receiver will be uninterrupted.

Schemes using temporal redundancy incur a 100% bandwidth overhead per stream. Further, to be effective, temporal redundancy must incur a delay on the received stream at least equal to the block separation period, which in turn must be at least as great as the period of loss of connectivity the technique is aiming to protect against. The additional delay, however, need not impact finger-to-eye delay if temporal redundancy is used only in the core network, i.e. does not extend to the end-user.

- *Loss in the application end-systems*. Loss due to buffer underflows and overflows can be prevented through careful design both of the network and the application end-systems, as discussed in Section 1.3.1.2.

Therefore, in practice, networks supporting video streaming services should typically be designed for very close to zero percent video packet loss. QOS mechanisms, admission control techniques and appropriate capacity planning techniques are deployed to ensure that no packets are lost due to congestion, with the only actual packet loss being due to layer 1 bit errors or network element failures. Where packet loss occurs, the impact of the loss should be reduced to acceptable levels using concealment techniques. Different concealment techniques may be employed in different parts of the network. For example, in the core of the network where bandwidth is relatively plentiful and diverse paths exist, spatial redundancy may be used to transport video across the core of a network from a source to distribution points; this would provide protection against both lower layer errors and against network element failures. Other techniques such as FEC could then be used from the video redistribution points to the video receivers, in order to provide protection against lower layer errors in the access network.

1.3.2.1.4 Video Streaming: Impact of Throughput

The bandwidth requirements for a video stream depend upon the video format, the encoder and the specific GOP structure. There are four main video formats used for IP-based video services:

- *Standard definition (SD)*. Standard definition format is the conventional format commonly used today for broadcast quality digital

video encoding including television services such as cable and satellite. The nomenclature used to refer to digital TV image definitions defines the number of vertical lines and whether the image is interlaced (i) or uses progressive scanning (p); two of the main SD formats are 480i and 576i:
  - 480i: 480 vertical lines (interlaced) × 720 horizontal pixels at 29.97 fps (NTSC-based, North America)
  - 576i: 576 lines by 720 pixels at 25 fps (PAL-based, Europe)

- *High definition (HD)*. High definition has at least twice the resolution (in terms of total number of pixels) as SD and is used for premium broadcast quality digital video encoding including television services. Two of the main HD formats are:
  - 720p: 720 × 1280 at 50 Hz and 60 Hz frame rates (progressive scan)
  - 1080i: 1080 × 1920 at 25 Hz and 30 Hz frame rates (interlaced)

- *Common interchange format (CIF)*. CIF is a low definition (LD) format targeted for broadband Internet video delivery applications, such as video conferencing. CIF has one quarter of the full resolution of an SDTV picture (i.e. frame height and width are both halved), which in turn is referred to as 4CIF. Two common 4CIF formats are:
  - 240i: 240 × 352 (NTSC-based)
  - 288i: 288 × 352 (PAL-based)

- *Quarter CIF (QCIF)*. QCIF is targeted for mobile handset video applications and has one quarter of the resolution of CIF. Two common QCIF formats are:
  - NTSC-based: 120 × 176
  - PAL-based: 144 × 176.

MPEG allows for streaming video to be encoded either as variable bit rate streams, where the quality of the resultant video is constant, or as constant bit rate streams where the quality of the resultant video is variable. The table in Figure 1.16 gives indicative average bit rates for LD, SD and HD video stream rates using MPEG-2 and MPEG-4 AVC.

McCann observed [MCCANN] that due to the evolution of video encoding implementations, from 1995 to 2002 there was an average

Format	MPEG-2	MPEG-4 AVC
LD QCIF	100–200 kbps	50–100 kbps
LD CIF	0–5–1 Mbps	0.25–0.5 Mbps
SD 4CIF	~3–4 mbps	~2–3 Mbps
HD	~15–20 Mbps	~10–15 Mbps

Figure 1.16 Typical broadcast quality video stream IP rates

of 15% improvement in MPEG-2 encoding efficiency and consequent reduction in stream bandwidth per year. The MPEG-2 bit rates shown in Figure 1.16 are nearing the end of this improvement cycle, whereas there is still scope for reduction in MPEG-4 bit rates.

It is further noted that larger GOP structures, which result in reduced bit rates for equivalent quality streams, have an impact both when considering the effect that packet loss has on the stream and on the channel change time for broadcast video services. Hence, when opting for larger GOP structures to reduce bandwidth, it should be understood that there might be a consequent trade-off in terms of visual quality and channel change time.

From a network design perspective, as discussed in Section 1.3.2.1.3, networks supporting streaming video should typically be designed for very close to zero percent video packet loss, and hence are designed to be congestionless from the perspective of the video traffic. This means that the available capacity for video traffic must be able to cope with the peak of the offered video traffic load. Further, in order to ensure that the service availability is maintained this peak load must be able to be supported without loss while maintaining the required delay and jitter bounds for the video traffic. The capacity planning required to achieve this is discussed in more detail in Chapter 5, Section 6.1.

As for VoIP, as discussed in Section 1.3.1.4, this does not mean that the available capacity for streaming video cannot be oversubscribed. Where oversubscription is used, admission control may be needed to limit the concurrent number of video streams that can be set up; admission control is discussed in more detail in Chapter 4.

1.3.2.1.5 Video Streaming: Impact of Packet Re-ordering

Many real-time video end-systems do not support the re-ordering of received frames, hence packet re-ordering effectively results in higher packet loss and should be avoided.

1.3.2.2 Video Conferencing

Video conferencing sessions are typically set up using the signaling protocols specified in ITU recommendation H.323 [H.323] or SIP. Whichever method is used to establish the connections, from an SLA perspective, the fundamental requirements and principles remain the same. Once the H.323 or SIP end points (also known as terminals) have agreed that they are willing to communicate with each other and have determined the media formats (codecs) that they will use, they then set up the logical channels through which the bearer (media streams such as VoIP and video) traffic itself will be transmitted. Typically, separate logical channels are used for audio and for video, and the setup of these logical channels consists of determining which particular UDP ports the RTP transported media streams will use.

The audio streams will typically use codecs such as those defined by the ITU G.71x/G72x standards. The SLA requirements with respect to delay, jitter, loss, throughput and in-sequence delivery are therefore the same as those already described for VoIP in Section 1.3.1.

The video formats and encoding used for video conferencing applications are less constrained than for broadcast quality video services. Codecs such as MPEG-2/H.262 or MPEG-4 AVC/H.264 are typically used; where bandwidth is constrained, lower definition (e.g. CIF or QCIF) and lower frame rates (e.g. 10 fps), potentially reduce the bandwidths required significantly compared to broadcast video services.

Although the bandwidth requirements for the video streams used for video conferencing may be less than for broadcast video, the delay and jitter requirements will depend upon the quality of the video conferencing service offered. With lower quality services, it may be acceptable for the video stream to have a less stringent delay commitment than the voice stream, in which case participants may experience a time lag between a remote user's audible words, and their associated lip movements. For higher quality services, where lip synchronization is required, one of the media streams may need to be delayed at the receiver, however, subjective studies have shown that the two streams do not have to be exactly matched. [STEINMETZ] shows that the synchronization errors that can be tolerated by human perception vary in different application scenarios; for video conferencing applications, a skew of less than 80 ms is below the limit of human perception. Therefore, even for higher quality services, it may be acceptable for the video stream to be delayed by up to 80 ms with respect to the audio stream, i.e. the end-to-end delay budget for the video stream can be up to 8 ms greater than for the audio stream. Hence, extrapolating from the G.114 targets for voice services, end-to-end (i.e. camera to eye) delay targets of ~250 ms are targeted for the video stream of video conferencing applications.

As for discrete voice and video services, in practice networks supporting video conferencing services should typically be designed for very close to zero percent packet loss for both the VoIP and video streams. QOS mechanisms and appropriate capacity planning techniques may be employed to ensure that no packets are lost due to congestion, with the only actual packet loss being due to layer 1 bit errors or network element failures. Where packet loss occurs, the impact of the loss on voice streams should be reduced to acceptable levels using concealment techniques. The loss rates tolerated for video conferencing are likely higher than those acceptable to broadcast video services, such that the complexity of the video loss concealment techniques may not be required.

1.3.3 Data Applications

QOE requirements for data application, which in turn drive network level SLAs, are less well defined than for voice or video applications. While there are multiple types of data applications that exist, from a QOS perspective they can be broadly divided into interactive data applications and applications that are targeted at data transfer with no requirements on interactivity. An example of an application that is targeted at data transfer, but has no requirements on interactivity, is a data backup application between data centers, where a defined averaged throughput is required in order to ensure that the backup completes within a determined time period.

Throughput focussed applications in general use TCP as their transport layer protocol, due to the reliability and flow control capabilities that it provides. The impact that different SLA metric parameters have on TCP session throughput is considered in Section 1.3.3.1. While there may be some throughput focussed applications that use UDP rather than TCP, such as the Trivial File Transfer Protocol (TFTP) [RFC783] for example, as UDP does not have any implicit reliability or flow control mechanisms these need to be built into the application implementation. Hence, a detailed knowledge of the specific application implementation is required in order to understand what impact different SLA metric parameters have on such UDP applications, which would need to be analyzed on a case-by-case basis.

Interactive data applications require that a transaction be completed within a certain period of time, to ensure that the attention of the end-user is maintained or so that they do not consider that a fault has occurred, for example. The impact that different SLA metric parameters have on interactive data applications is considered in Section 1.3.3.2.

1.3.3.1 Throughput Focussed TCP Applications

In order to understand the impact that metrics such as network delay and loss have on TCP [RFC793], we first need to understand the basic principles of TCP operation. TCP aims to provide a reliable and efficient transport layer protocol on top of IP. Four key mechanisms underlay TCP:

- bidirectional session establishment
- positive acknowledgment with retransmission
- sliding acknowledgment window for flow control between the sender and receiver
- congestion control, for dealing with loss which occurs between the sender and receiver.

The following sections provide an overview of these mechanisms.

1.3.3.1.1 Bidirectional Session Establishment

TCP is a connection oriented transport protocol, which establishes a session between two TCP end-systems before any data can be transferred. TCP session establishment relies on a "three-way handshake." Consider two TCP end-systems, A and B, where A is the initiator of a TCP session to B;

1. A first sends a TCP segment to B with the synchronize bit (SYN) set; this indicates that it is the first segment of the three-way handshake. A segment is the TCP data unit transported in an IP packet.
2. To progress the establishment of the TCP session, B responds by sending a segment back to A with both the SYN and acknowledgment (ACK) bits set.
3. The final segment in the handshake is another ACK, which is sent by A to B to confirm that the session has been successfully established.

During session establishment, a number of parameters are negotiated including the maximum segment size (*MSS*) and the window size, which will be discussed in the following sections, and also whether Explicit Congestion Notification will be used (see Chapter 2, Section 2.3.4.4).

1.3.3.1.2 Positive Acknowledgment with Retransmission

TCP uses a technique known as "positive acknowledgment with retransmission" in order to ensure reliable and efficient data transfer between the TCP end-systems, even if the underlying network may be unreliable. In the context of TCP, positive acknowledgment with retransmission requires that the receiver sends an acknowledgment (ACK) back to the sender on receipt of a TCP segment. In the simplest possible model, the sender keeps a record of each segment it sends and waits for an ACK for the preceding segment, before sending the next. The sender also starts a retransmission timer when it sends a segment; if no ACK is received before the timer times-out the sender retransmits the segment, presuming that the one previously sent was lost.

TCP is a full duplex transport protocol, which means that a single TCP session can support data transfer in both directions. Hence, ACKs from one TCP end-system to another can be piggybacked with segments being sent in the opposite direction.

1.3.3.1.3 Sliding Acknowledgment Window

TCP extends the basic positive acknowledgment with retransmission model by adding the concept of a sliding acknowledgment window, which allows the sender to transmit multiple packets to the receiver before waiting for an ACK. The sliding window acknowledgment scheme is used by TCP for flow control, i.e. adjusting the rate of transmission by the sender so that it does not exceed the receiver's capacity to accept data.

Each TCP end-system advertises the maximum window size (*awnd*) they are prepared to accept to their session-peer within the TCP header. A window aperture or size of X packets means that the sender can have up to X unacknowledged segments outstanding; if the sender has received an ACK for segment Y, then the sender can transmit up to segment Y + X before receiving the ACK for segment Y + 1. There can be a number of packets in transit between sender and receiver with the concept of a sliding window, hence in order to track lost and duplicate segments TCP assigns a sequence number to each segment. If one (or more) of the packets in transit are dropped or resequenced it will appear to the receiver as an out-of-order segment; this causes the receiver to immediately generate an ACK (called a duplicate ACK) for the last segment successfully received in sequence.

1.3.3.1.4 Congestion Control

TCP also implements congestion control, to avoid network "congestion collapse" [JACOBSON] by controlling TCP sessions to adapt to the network and conditions in order to try to maximize throughput for each session. This also maintains fairness between sessions.

To achieve this, the TCP window size is not statically defined but rather a number of control congestion techniques [RFC2581] are used

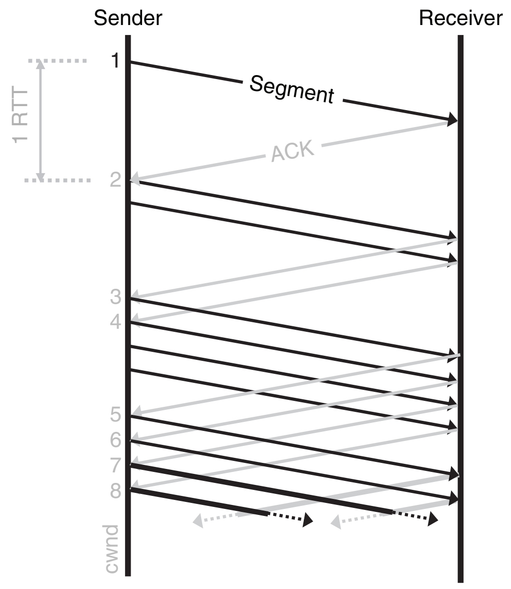

Figure 1.17 TCP slow start

to adjust the window size dynamically. The main techniques used are *slow start, congestion avoidance, fast retransmit* and *fast recovery*:

- *Slow start.* The slow start algorithm aims to control the rate at which a sender transmits segments into a network at the start of a session, in order to reduce the probability that the session will send too many segments and contribute to congestion at a network element on the session's path. To achieve this, slow start applies a constraint to the window size, called the congestion window (*cwnd*). When a new session is established, *cwnd* is set to one segment; for each ACK that is received, *cwnd* is increased by one segment. A sender can send up to whichever is the minimum of *awnd* and *cwnd* (i.e. MIN[*awnd*, *cwnd*]), before another ACK is received. For example, if a sender in slow start with *cwnd* set to one sends one segment; when it receives the first ACK it increases *cwnd* to two and sends two more segments; when each of those two segments are acknowledged, *cwnd* is increased to four as shown in Figure 1.17.

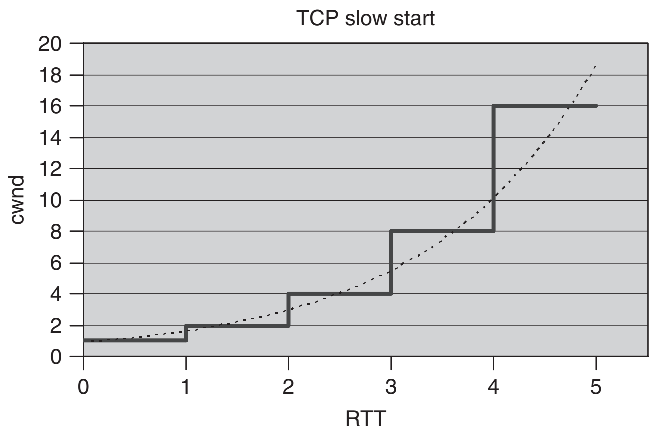

Figure 1.18 TCP slow start

Therefore, slow start provides exponential growth of *cwnd* with RTT as shown in Figure 1.18. You might then question why slow start is so called? The reason is because it is slow relative to the original implementations of TCP, which had no concept of *cwnd* and sent using the maximum negotiated window size at the start of the session.

- *Congestion avoidance*. At some point, the capacity of an intermediate network element between sender and receiver may be exceeded, i.e. congestion occurs and packets may be dropped as a consequence. TCP (without support for ECN as discussed in Chapter 2, Section 2.3.4.4) effectively treats the network as a "black box," in that it does not rely on any explicit network behaviors when performing flow control, in order to determine the status of available network bandwidth and whether congestion has occurred, TCP relies on TCP timeouts or the reception of duplicate ACKs to determine implicitly when packets are dropped. When packet drops within a session are determined, TCP reacts by invoking the following algorithm, which uses an additional variable called the slow start threshold (*ssthresh*):

1. *ssthresh* is set to whichever is the greater of 2 segments or one-half of the window size before the congestion occurred (i.e. MAX[2, MIN[*awnd, cwnd*]]). The *cwnd* is then halved for every subsequent

loss, hence if loss continues the rate of transmission effectively decreases exponentially.

In addition, if the congestion is indicated by a timeout, *cwnd* is set to one segment.

2. When the next ACK is received:
   - If *cwnd* is less than or equal to *ssthresh*, slow start is invoked as described above until the window size equals *ssthresh*, i.e. half of the window size when the congestion occurred.
   - When *cwnd* is greater than *ssthresh*, invoke the following congestion avoidance algorithm.

     The congestion avoidance algorithm defines that each time an ACK is received *cwnd* is incremented by *segment_size/cwnd*. For example, if a sender in slow start with *cwnd* set to one sends one segment, when it receives the first ACK it increases *cwnd* to two and sends two more segments; when the next two segments are acknowledged, *cwnd* is increased to three as shown in Figure 1.19.

   The increase in *cwnd* is limited to at most one segment during each RTT irrespective of how many ACKS are received in

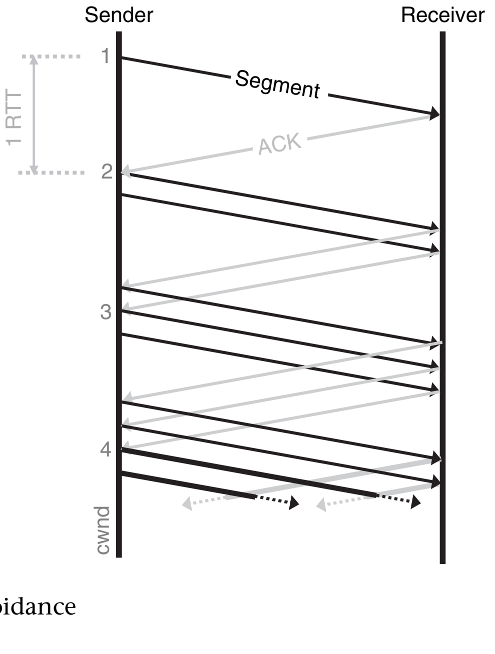

Figure 1.19 TCP congestion avoidance

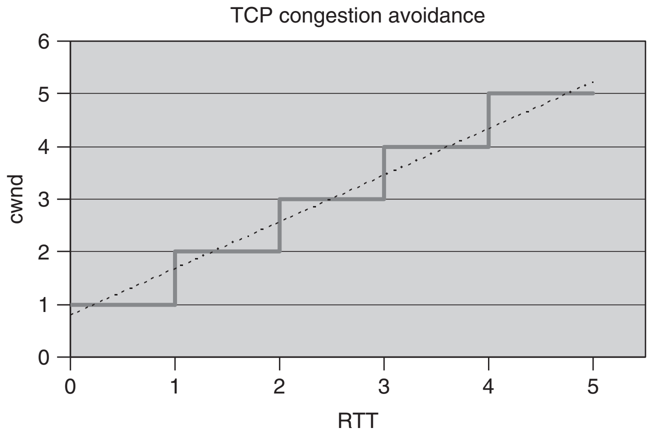

Figure 1.20 TCP congestion avoidance

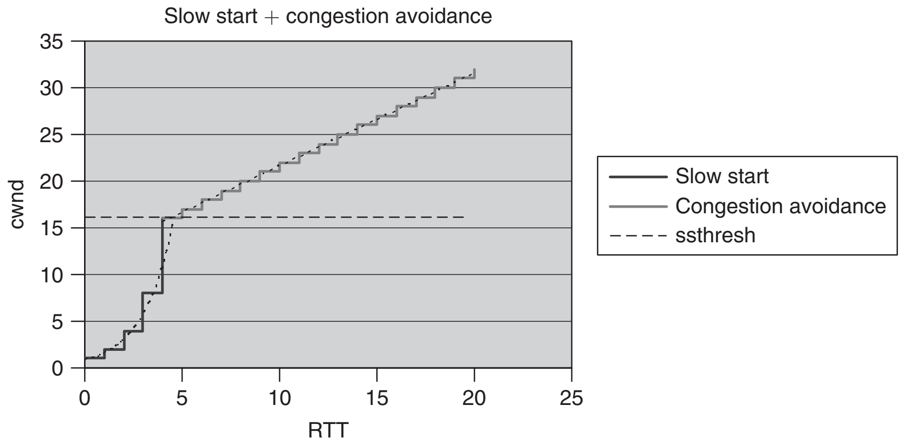

Figure 1.21 Slow start and congestion avoidance example

that interval, whereas slow start increases *cwnd* by the number of ACKs received within a RTT. Therefore, congestion avoidance provides linear growth of *cwnd*, compared to exponential growth with slow start.

Consider the example shown in Figure 1.21; at time $t = 0$, an established TCP session experiences a timeout when *cwnd* = 32 segments;

ssthresh is set to 16 segments and *cwnd* is reduced to 1 segment. *cwnd* then increases with slow start until it equals *ssthresh*, then congestion avoidance is invoked. As can be seen from Figure 1.21, *cwnd* increases exponentially with RTT until it equals *ssthresh*; above *ssthresh cwnd* increases linearly with RTT.

- *Fast retransmit*. Fast retransmit is a performance enhancement to the previously described mechanisms, which determines the sender's behavior when duplicate ACKs are received. When a sender receives a duplicate ACK, it does not know whether the ACK was caused by a lost segment or a re-ordering of segments, hence it waits for a small number of duplicate ACKs to be received before reacting. If two or less duplicate ACKs are received in a row it is assumed that segment re-ordering has occurred. If three or more duplicate ACKs are received, it is taken as an indication that a segment has been lost, and fast retransmit defines that in this case the sender should retransmit the missing segment, without waiting for the retransmission timer to expire.
- *Fast recovery*. Fast recovery is a further performance enhancement, which allows higher throughput under moderate congestion. Fast recovery defines that after fast retransmit resends the missing segment, congestion avoidance should be performed rather than slow start.

Support for the various congestion control techniques in TCP stacks has evolved over time and there is a generally used naming scheme, which derives from the BSD (Berkeley UNIX) TCP stack implementation, and which provides a taxonomy for TCP stack evolution:

- 4.2BSD (1983) was the first widely available release of TCP/IP.
- *Tahoe*: the 4.3BSD Tahoe release (1988) incorporates slow start, congestion avoidance and fast retransmission.
- *Reno*: the 4.3BSD Reno release (1990) added support for fast recovery.
- *Vegas*: the Vegas TCP stack (1994) added support for the additional enhancements described in [BRAKMO], which aim to improve TCP throughput over previous stacks.

- *NewReno*: the NewReno TCP stack (1999) adds support for the enhancements to the fast recovery algorithm described in [RFC3782].

More recently, [RFC 3390] specified an increase in the permitted upper bound for TCP's initial window size from one or two segment(s) to between two and four segments. This change reduces the number of RTTs required to complete some transactions; many email and web page transactions are less than 4 kbytes, hence the larger initial window would reduce the data transfer time to a single RTT.

There are additional subtleties to the implementations of TCP – see [STEVENS, RFC2581] – however, the details provided above are sufficient to understand the impacts that network characteristics such as delay and loss have on TCP, and which are discussed in more detail in the following sections.

1.3.3.1.5 TCP: Impact of Delay

For TCP, the important delay metric is the RTT between TCP end-systems. The maximum number of unacknowledged segments that a TCP sender can have outstanding is limited by the window size. Therefore, for a particular RTT between sender and receiver, the maximum possible TCP session throughput will be determined by *window_size * MSS/RTT,* where *MSS* is the maximum segment size. This is the amount of data that has been sent, but not yet acknowledged and is commonly referred to as the TCP "flight size" [RFC2581] or the "pipesize." Hence, the TCP session throughput is inversely proportional to the RTT.

[MATHIS] shows that maximum theoretical attainable TCP throughput for a single session varies as a function of RTT and packet loss using the following relationship, where p is the probability of packet loss:

$$TCP\_throughput = \frac{MSS}{RTT \times \sqrt{p}}$$

The graph in Figure 1.22 uses this relationship to plot how the theoretical maximum attainable TCP throughput for a single TCP session varies as a function of RTT for a TCP maximum segment size (MSS) of 1460 bytes.

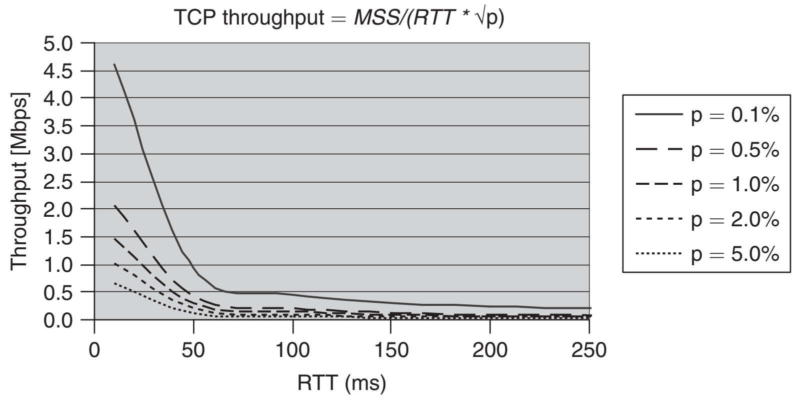

Figure 1.22 TCP throughput as a function of RTT

It is noted that the attained TCP throughput will in practice depend upon a number of additional factors, which vary network-by-network, including:

- the life span of the TCP sessions; long-lived sessions will have more opportunity to open up their maximum window sizes than short-lived sessions
- if congestion occurs:
  - the specific behavior of the participating end-systems in the presence of congestion; for TCP-based applications, this is dependent upon the TCP stacks used
  - the dropping behavior of any routers along the path in the presence of congestion, e.g. tail drop or RED (see Chapter 2, Section 2.2.4.2).

Additionally, it can be seen from Figure 1.22 that even if the attained throughput is close to the theoretical maximum, that at low RTTs and low probabilities of packet loss, the session throughput is limited to a few megabits per second. For some high bandwidth applications these limits can be overly constraining; one such application is Grid computing [BAKER], where geographically distributed computing

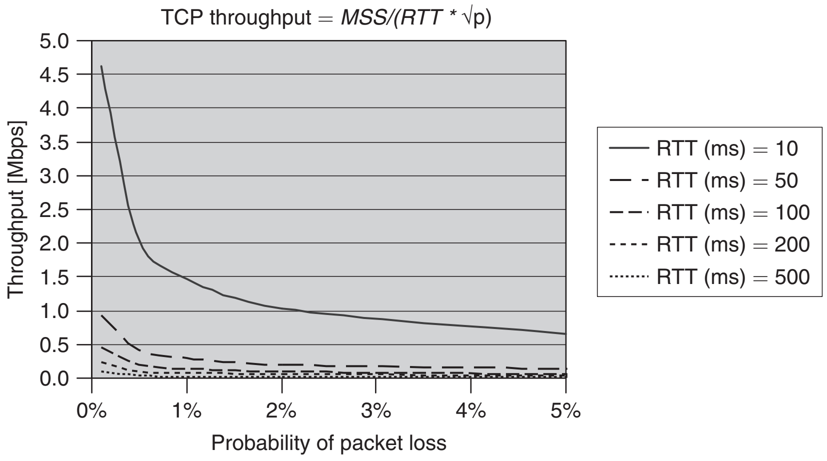

Figure 1.23 TCP throughput as a function of packet loss

resources are networked to act as a single unified computer. Consequently, there have been a number of efforts, although none of which are yet widely deployed, to enhance TCP in order to achieve much greater throughput with a single session; two such efforts are "Fast TCP" [CHENG] and "HighSpeed TCP" [RFC3649].

1.3.3.1.6 TCP: Impact of Delay-jitter

Jitter has no explicit impact on TCP; jitter only has an impact on TCP in that it is a component of delay, which can impact throughput as discussed in Section 1.3.3.1.5.

1.3.3.1.7 TCP: Impact of Loss

As discussed in Section 1.3.3.1.3, TCP implicitly has mechanisms to ensure that packets dropped are resent; however, [MATHIS] shows theoretically that maximum TCP throughput decreases as an inverse of the square root of the probability of packet loss. The graph in Figure 1.23 uses this relationship to plot how the theoretical maximum attainable TCP throughput for a single TCP session varies as a function of packet loss for a TCP maximum segment size (MSS) of 1460 bytes.

As previously noted in Section 1.3.3.1.5 there are a number of factors which will impact the attained TCP throughput in practice.

1.3.3.1.8 TCP: Impact of Throughput

The previous sections have described how the achieved throughput for a TCP session is dependent upon the probability of packet loss and the achieved RTT, as well as a number of practical factors. In addition achieved throughput for a TCP session will obviously be gated by the available capacity on the path between the source and destination. Hence, it is important to note that the actual achieved TCP throughput for a single TCP session may be significantly less than the contracted access bandwidth provided for a service.

1.3.3.1.9 TCP: Impact of Packet Re-ordering

Most deployed TCP implementations will support fast retransmit behavior and hence will interpret the receipt of three consecutive duplicate ACKS as an indication of packet loss and will retransmit the next packet and slow down their rate of sending by invoking the congestion avoidance algorithm. Therefore, re-ordering packets within a TCP stream can have a significant impact on TCP throughput. [LAOR] shows that for TCP traffic with a 0.04% rate of packet re-ordering, achieved application throughput can be reduced to 74% compared to the throughput achieved with no packet re-ordering.

Hence, packet re-ordering should be avoided due to the potential impact on TCP throughput.

1.3.3.2 Interactive Data Applications

Interactive applications depend on providing responses to an end-user in real-time. As the specific implementations of interactive data applications can vary, the impact that network characteristics such as delay have on them can also vary. Hence, it is not possible to provide definitive guidance, but rather we consider the main factors that impact the SLA requirements in support of interactive data applications.

The response time targets for such interactive applications are dependent upon human factors; [DOHERTY] show the economic value of rapid response times to interactive applications. Robert B. Miller's definitive paper from 1968 (when all computers were mainframes) on "Response Time in Man-Computer Conversational Transactions" described three response time thresholds for human attention, which are still generally accepted targets today:

- A response time of less than ~0.1 second is the target where applications need to give the user the feeling that the system is reacting instantaneously. This delay bound target is supported by more recent research in a variety of fields aimed at determining the delay above which performance for interactive applications becomes impaired [G.114, BAILEY, MACKENZIE].
- A response time of less than ~1.0 second is the target where the applications need to keep the user's flow of thought uninterrupted, although the user will notice the delay.
- A response time of less than 10 seconds is the target where the applications need to keep the user's attention focussed on the dialogue; for longer delays, users will want to perform other tasks while waiting for the computer to finish.

  This target was also supported by Peter Bickford's 1997 paper [BICKFORD], which reported research in which half the users abandoned web pages after a wait of 8.5 seconds; the "8-second rule" subsequently become a universal rule of web site design.

These targets relate to the time to complete a user transaction – that is, the time between a user action, and the user receiving the response to their action. For client/server applications which require a network transaction, network delay is but one aspect of the total transactional delay to which these targets refer, which may be comprised of the following components:

- *Client-side processing delays*. The user's system may perform some preprocessing before starting the network transaction, and may perform additional processing on receiving the response from the server.
- *Server-side processing delays*. The server will need to perform some processing before the response can be sent to the user. Some applications may involve a number of distributed server transactions as part of the processing initiated by user request.

- *Network delays*. Network delays will be incurred in sending the request from the client to the server and in sending the response from server to client. Further, some "chatty" applications may require several network transactions between the client and server for a single user transaction. For such applications, relatively small increments in network latency can have a noticeable effect on end-user response times, or conversely, very small network delays may be required in order to achieve the application response time targets.

Hence, for an application targeting a 1-second response time, even if the network RTT may be well below a second, the user response time may still exceed the one-second target. Hence, a good understanding of the specific application behavior is required to understand what impact different network delays have on the application and to be able to translate application response time targets into the corresponding network RTT targets.

Assume, as per the timeline shown in Figure 1.24 for example, that a business-critical interactive application with a 1 second response

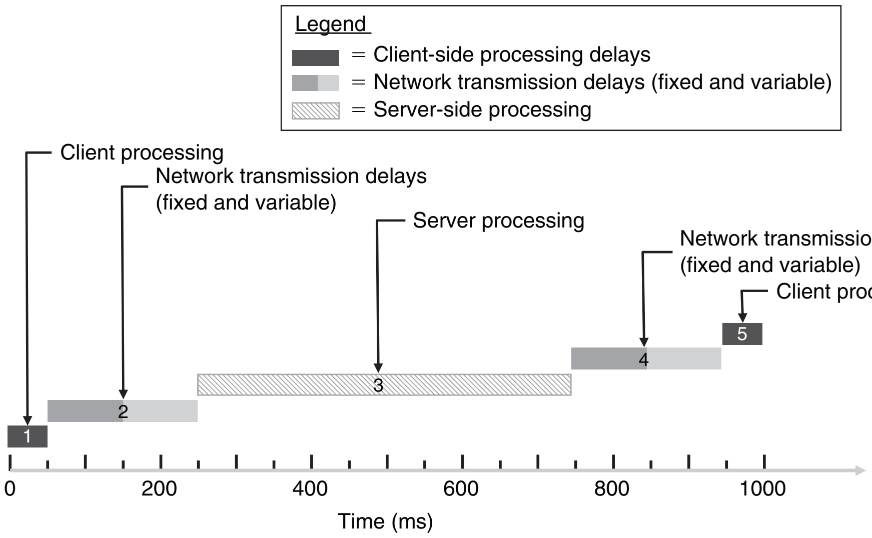

Figure 1.24 Delay components: example interactive data application #1

time target uses a single network transaction (e.g. an HTTP GET) per user transaction, using a minimally sized request (a single packet), with a total of 100 ms of client-side processing (actions 1 + 5) and 500 ms of total server-side processing (action 3). To meet the application response time target requires a network RTT of approximately 400 ms (actions 2 + 4) or less.

If by comparison, as per the example timeline shown in Figure 1.25, an application with the same total client-side, and server-side processing delays, but which instead required two network transactions (a DNS query and an HTTP GET for example) per user transaction, a network RTT of approximately 200 ms or less would be required in order to meet the target.

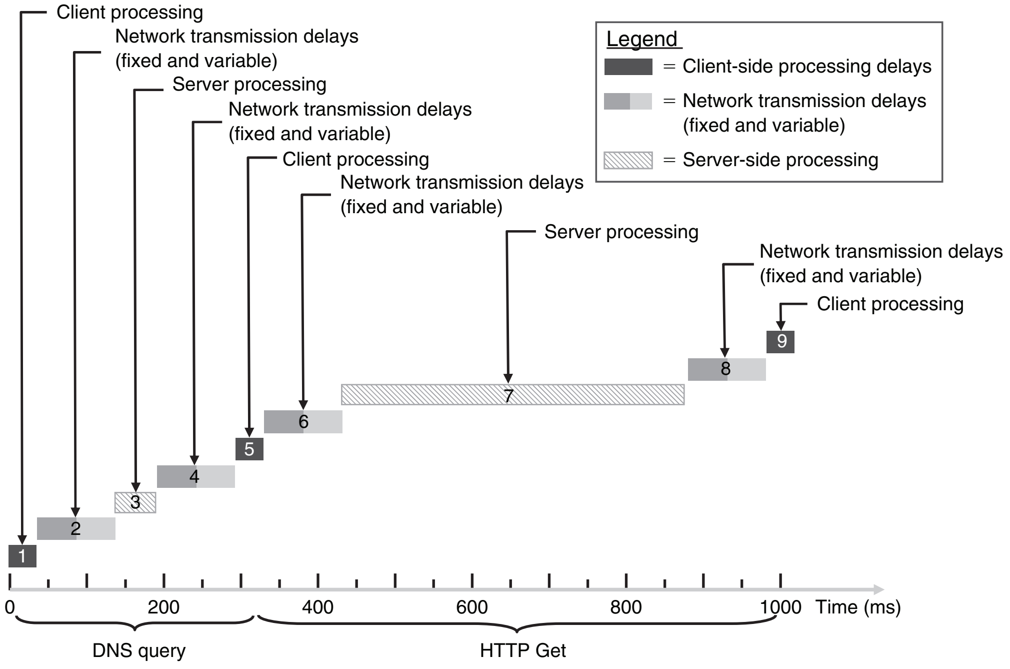

Figure 1.25 Delay components: example interactive data application #2

A badly designed application implementation may impose RTT requirements on the network that are not viable; in this case re-engineering of the application should be considered. Application re-engineering may require re-writing the application to reduce the number of network transactions, or relocating the server to be closer to the client, thereby reducing the network RTT.

Jitter has no explicit impact on interactive data applications; jitter only has an impact on TCP in that it is a component of network delay. Network loss and packet re-ordering can have an impact on interactive data applications in that lost or re-ordered packets may need to be retransmitted which may probabilistically increase the network component of the total transaction delay. The impact of packet loss and resequencing will depend upon the characteristics of the transport layer protocol that is used; the impact of packet loss and resequencing on TCP is discussed in Section 1.3.3.1. For UDP-based interactive data applications, a detailed knowledge of the specific application implementation is required in order to understand the impact of packet loss and resequencing; this would require analysis on an application-by-application basis.

1.3.3.3 On-line Gaming

Multiplayer on-line or networked games are the most popular form of a type of application known as Networked Virtual Environments (NVEs); other uses of NVEs include military simulation. Users in NVEs, who may be in geographically separate locations, interact with each other in a virtual world in real-time. The IEEE Distributed Interactive Simulation (DIS) [IEEE1278] standard covers NVE; however, this is not generally used by the software vendors that produce on-line games who instead use proprietary implementations. The variation in such proprietary implementations means that it is not possible to provide definitive guidance on SLA requirements to support on-line gaming applications, but rather we review the current research on this subject.

Although there are different types of real-time on-line games – the most common game types being: First Person Shooter (FPS), Real-Time Strategy (RTS) and Multiplayer On-line Role-Playing Game

(MORPG) – most use a client-server architecture, where a central server tracks client state and hence is responsible for maintaining the state of the virtual environment. The players' computers are clients, unicasting location and action state information to the server, which then distributes the information to the other clients participating in the game. Most implementations use UDP as a transport protocol.

Most on-line gaming implementations have evolved to work over the public Internet and have bandwidth requirements of less than 64 kbps and in-built mechanisms to deal with packet loss. However, it is noted that these bandwidth requirements may increase over time, with the prevalence of higher bandwidths available to end-users due to broadband access. In addition, some games provide the capability to tweak various network parameters, which can have a significant impact on their bandwidth requirements.

It is commonly cited that low network delay is a requirement of on-line gaming applications; players who experience higher delays to/from the server than others may experience a relative "lag" in play as they receive information from the server later than lower delay users, and similarly the server receives information from them later than from the lower delay user. Consequently, users with lower RTTs may have a game-playing advantage. In terms of setting a bound on the acceptable RTT for on-line gaming, research into FPS games suggests that with delays above 100–250 ms, gamers are deterred from playing and/or their playing performance is inhibited [HENDERSON1, ARMITAGE, PANTEL], although different types of games might have differing requirements. These findings are supported by research in a variety of fields aimed at determining the delay above which performance for interactive applications becomes impaired [G.114, BAILEY, MACKENZIE].

A more recent study by [HENDERSON2] examined the impact that delay has on user behavior, rather than impact on game-play, concluding that high network delay may dissuade a user from joining a particular game server. However, when experiencing high delay after having joined a game server, even when players could notice the delay and their game-play performance was degraded, they were not inconvenienced to the extent that they would leave the server.

This may indicate that players are willing to tolerate higher levels of delay than previous research indicated. In addition, a number of techniques for lag compensation have been shown to be successful [BERNIER].

Research into RTS games [SHELDEN] suggests that RTTs of up to 500 ms have minimal effect on the end-users; even though higher latencies were noticeable to the user they had negligible effect on the outcome of the game; this is attributed to the nature of RTS game play, which is focussed more on strategy, rather than real-time aspects.

1.4 Marketed SLAs versus Engineered SLAs

In practice, there may be a distinction between *marketing SLAs* and *engineering SLAs*. Marketing SLAs apply in SP environments, are contracted between the SP sales channel and their customers, and are aimed to be simple for the customer to understand, competitive for the targeted customer market segment and easy to report against. Marketing SLAs may also define the bounds that represent SLA violation, together with the service-credits/refunds that apply in the presence of such violation. Marketing SLAs do not really have any context in enterprise network environments.

Engineering SLAs apply in both network SP and enterprise environments. In an SP context engineering SLAs are contracted between the SP engineering and support team and their sales channel and are aimed to support the requirements of the applications of the targeted customer market segment; SPs will not normally disclose their engineering SLAs. In an enterprise context, engineering SLAs are contracted by the network engineering and support teams to the enterprise and are aimed to support the requirements of the enterprise's business critical applications. Engineering SLAs define the bounds used in designing and operating the network and are not visible to the end customer. They are necessarily more stringent than their equivalent marketing SLA, to mitigate the risks of paying service-credits/refunds. As the focus of this book is on engineering, the preceding sections and the rest of this book focus on engineering SLAs; how an SP chooses to translate an engineering SLA commitment to a marketing SLA is a choice for the particular service provider.

1.4.1 End-to-End SLAs vs Segmented SLAs

For ease of understanding, marketing SLAs are normally defined end-to-end – that is, from customer premises equipment (CPE) to CPE – in the context of services where the SP manages the CPE. End-to-end SLAs, however, often need to be applicable across an amalgamation of geographic locations and possibly link speeds; therefore, their definition needs to be loose enough to encompass the worst case.

Consequently, to ease the problem of engineering the network, engineering SLAs are most commonly defined in a segmented manner: the access links are designed to meet an edge engineering SLA, while the backbone is designed to meet a core engineering SLA. A small and well-defined set of segmented SLAs could be used to support a larger and more complex set of marketing SLAs. The segmented approach also maps well to use of active SLA monitoring, where segmentation can aid scaling of the deployment of an active SLA probing system (see Chapter 5, Section 5.3).

1.4.2 Inter-provider SLAs

When considering services which span multiple providers' networks, it might be assumed that from a SLA perspective, the end-to-end SLA for a class (assuming there are congruent classes between the two providers) is a simple aggregation of the individual SLAs offered by the providers, e.g. SP A offers a loss rate of 1%, SP B offers a loss rate of 2%, hence the end-to-end loss rate is $1 - (0.98 * 0.97) = \sim 3\%$. However, in practice, the marketing SLA definitions for the two providers are likely to be very different: SP A might define packet-loss by sending end-to-end probes every minute and then averaging daily, while SP B might send probes hourly and average monthly. This illustrates how much scope there is for ambiguity with marketing SLAs.

Marketing SLAs aside, as SPs will actually design their networks to satisfy engineering SLAs with the objective that the target applications for each class of service they offer will actually work across their own network. Hence, although a VoIP service might be able to be supported across SP A and across SP B individually, there is nothing to assure that it will work end-to-end across both SP A and SP B's networks in series. Worse still, the VoIP service might work initially, because both SPs' networks are lightly loaded; however, when their networks become more heavily loaded, the VoIP service may fail, even though neither SP is exceeding their individual SLAs. This applies not just to delay but also to jitter, loss, throughput and availability.

A benefit of using a solution where SPs have cooperated in providing an inter-provider service is that the network has been engineered to support the target applications end-to-end across both providers, and there is some recourse for the end customer if this does not work. There are some current standards efforts, which are looking to formalize the definitions of inter-provider services [IPSPHERE].

1.5 Intserv and Diffserv SLAs

For completeness, Intserv and Diffserv both have their respective SLA terminologies; however, in practice these terminologies are rarely used in explicit SLA definitions:

- *Diffserv*. SLAs in Diffserv [RFC2475 updated by RFC3260] (Diffserv is described in Chapter 2, Section 2.3.4) are defined in terms of the Service Level Specification (SLS) and Traffic Conditioning Specification (TCS):
  - A SLS is a set of parameters and their values which together define the service offered to a traffic stream by a Diffserv domain. The SLS is effectively the engineering SLA that a service has been designed to support.
  - A TCS is a set of parameters and their values, which together specify a set of traffic classifier rules and a traffic profile. A TCS

is an integral element of an SLS. The TCS effectively defines the traffic (both in terms of classification and traffic profile) for which the SLS is committed.

- Diffserv also defines the concept of "Per-Domain Behaviors" (PDBs) which can be considered a definition of the "black box" forwarding behaviors experienced by a class of packets across a differentiated services network, and can be considered to be the Diffserv definition of the end-to-end engineering SLAs described in Section 1.4.

  Only a single PDB has been defined and that is the lower-effort PDB.

- *Intserv*. With Intserv/RSVP [RFC 2210], SLAs are defined in terms of the QOS service level (either Guaranteed Service [RFC 2212] or Controlled-Load Service [RFC 2211]) and the traffic specifier describing the level of traffic for which the service is assured. See Chapter 4, Section 4.4 for more details on Intserv.

References[6]

[802-2001] IEEE standard for local and metropolitan area networks: overview and architecture

[ARMITAGE] G. Armitage, An experimental estimation of latency sensitivity in multiplayer Quake 3, in *Proceedings of the 11th IEEE International Conference on Networks (ICON 2003)*, Sydney, Australia, September 2003

[BAILEY] R. W. Bailey, *Human Performance Engineering – Using Human Factors/Ergonomics to Achieve Computer System Usability*, Prentice Hall, Englewood Cliffs, NJ, USA, second edition, 1989

[BAKER] Mark Baker, Rajkumar Buyya, and Domenico Laforenza, Grids and grid technologies for wide-area distributed computing, *Software – Practice & Experience*, Volume 32, Issue 15, December 2002, pp. 1437–1466

[BERNIER] Y. W. Bernier, Latency compensating methods in client/server in-game protocol design and optimization, in *Proceedings of the 15th Games Developers Conference*, San Jose, CA, USA, Mar. 2001

[BICKFORD] Peter Bickford, Worth the wait? contribution to the Human Interface On-line column of Netscape's View Source, October 1997. Available at: http://developer.netscape.com/viewsource/bickford_wait.htm

[BRATSKMO] L. S. Bratskmo and L. L. Peterson, TCP vegas: End to end congestion avoidance on a global internet, *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, 1995

[BT.500] ITU-R Recommendation BT.500-11, 2002-06, Methodology for the Subjective Assessment of the Quality of Television Pictures

[CHIMENTO] P. Chimento, J. Ishac, Defining Network Capacity, draft-ietf-ippm-bw-capacity-01 (work in progress)

[DICK] M. Dick, O. Wellnitz, L. Wolf, Analysis of factors affecting Players' Performance and Perception in Multiplayer Games, *NetGames'05*, Hawthorne (NY), U.S.A.

[DOHERTY] Walter J. Doherty, Ahrvind J. Thadani, The Economic Value of Rapid Response Time, November, 1982. http://www.vm.ibm.com/devpages/jelliott/evrrt.html

[DUFFIELD] N.G. Duffield, P. Goyal, A.G. Greenberg, P.P. Mishra, K.K. Ramakrishnan, Jacobus E. van der Merwe, Resource management with hoses: point-to-cloud services for virtual private networks, *IEEE/ACM Transactions on Networking*, November 2002

[EBU] http://www.ebu.ch

[ETSI Ts 102 034] Draft Technical Specification ETSI TS 102 034 V<0.0.8> (2004-05-05) Digital Video Broadcasting (DVB) Transport of DVB Services over IP

[CHENG] Cheng Jin et al., FAST TCP: from theory to experiments, *IEEE Network*, 19(1):4–11, January/February 2005

[FRANCOIS] Pierre Francois, Clarence Filsfils, John Evans and Olivier Bonaventure, Achieving subsecond IGP convergence in large IP networks, *ACM SIGCOMM Computer Communication Review*, Vol. 35, Issue 3 (July 2005), pp. 35–44

[G.107] ITU-T Recommendation G.107 (03/2005), The E-model, a computational model for use in transmission planning

[G.114] ITU-T Recommendation G.114, One-way transmission time, International Telecommunication Union, Geneva, Switzerland, Sep. 2003

[G.826] ITU-T Recommendation G.826, End-to-end error performance parameters and objectives for international, constant bit-rate digital paths and connections, International Telecommunication Union, Geneva, Switzerland, Dec. 2002

[H.323] ITU-T Recommendation H.323: Packet-Based Multimedia Communications Systems, International Telecommunication Union, Geneva, Switzerland, Feb. 1998

[HENDERSON1] T. Henderson, Latency and user behaviour on a multiplayer game server, in *Proceedings of the 3rd International Workshop on Networked Group Communication (NGC)*, pp. 1–13, London, UK, Nov. 2001

[HENDERSON2] Tristan Henderson and Saleem Bhatti, Networked games – a QoS-sensitive application for QoS-insensitive users?, *Proceedings of the ACM SIGCOMM workshop on Revisiting IP QoS*, pp. 141–147, Karlsruhe, Germany, August 2003. ACM Press

[IEEE1278] Institute of Electrical and Electronic Engineers, 1278.2-1995, IEEE Standard for Distributed Interactive Simulation – Communication Services and Profiles IEEE, New York, NY, USA, Apr. 1996

[IPFRR] M. Shand, S. Bryant, IP Fast Reroute Framework, Internet Draft, draft-ietf-rtgwg-ipfrr-framework (work in progress).

[IPPM] http://www.ietf.org/html.charters/ippm-charter.html

[IPSPHERE] www.ipsphereforum.org

[J.144] ITU-T Recommendation J.144 (03/2004), Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference

[JACOBSON] JACOBSON, V, Congestion avoidance and control, In *Proceedings of SIGCOMM '88* (Stanford, CA, Aug. 1988), ACM

[LAOR] Michael Laor, Lior Gendel, The Effect of Packet Re-ordering in a Backbone Link on Application Throughput, *IEEE Network,* Vol. 16, No. 5, pp. 28–36, 2002.

[MACKENZIE] I. S. MacKenzie and C. Ware, Lag as a determinant of human performance in interactive systems, in *Proceedings of the CHI '93 Conference on Human factors in computing systems*, pp. 488–493, Amsterdam, The Netherlands, Apr. 1993

[MATHIS] Matthew Mathis, The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm, *Computer Communication Review*, July 1997

[MCCANN] Ken McCann, DVB + MPEG-4 = New Options for Baseband Systems, *DVB World Conference 2002* – Dublin 6/7 March 2002. Available at: http://www.zetacast.com/Zeta_Publications.htm

[MPEG] ISO/IEC JTC1/SC29 WG11. More information available at: http://www.chiariglione.org/mpeg/

[MPEG-2] ISO/IEC 13818: Generic coding of moving pictures and associated audio information (MPEG-2)

[MPEG-4] ISO/IEC 14496-10:2004, Coding of audio-visual objects – Part 10: Advanced Video Coding

[NIELSEN] Jakob Nielsen, Usability Engineering, Morgan Kaufmann, 1994. Also on-line at: http://www.useit.com/papers/responsetime.html

[P.800] ITU-T Recommendation P.800 (08/96), Methods for subjective determination of transmission quality

[P.862] ITU-T Recommendation P.862 (02/2001) Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs

[PANTEL] L. Pantel and L. C. Wolf, On the impact of delay on real-time multiplayer games, in *Proceedings of the 12th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pp. 23–29, Miami Beach, FL, USA, May 2002

[PRO-MPEG] Pro-MPEG Forum Code of Practice #3 – release 2: Transmission of Professional MPEG-2 transport streams over IP networks, Professional-MPEG Forum, www.pro-mpeg.org

[PWE3] http://www.ietf.org/html.charters/pwe3-charter.html

[RFC768] J. Postel, User Datagram Protocol, August 1980

[RFC783] K. Sollins, The TFTP protocol (revision 2), RFC 783, July 1992

[RFC791] J. Postel, Internet Protocol, RFC 791, September 1981

[RFC793] J. Postel, Transmission Control Protocol, RFC 793, September 1981

[RFC2210] J. Wroclawski, The Use of RSVP with IETF Integrated Services, RFC2210, September 1997

[RFC2211] J. Wroclawski, Specification of the Controlled-Load Network Element Service, RFC2211, September 1997

[RFC2212] S. Shenker et al., Specification of Guaranteed Quality of Service, RFC2212, September 1997

[RFC2250] D. Hoffman et al., RTP payload Format for MPEG1/MPEG2 video, RFC 2250, January 1998

[RFC2326] H. Schulzrinne, A. Rao, R. Lanphier, Real-time Streaming Protocol (RTSP), RFC 2326, April 1998

[RFC2330] V. Paxson et al., Framework for IP Performance Metrics, May 1998

[RFC2343] M. Civanlar, G. Cash, B. Haskell, RTP Payload Format for Bundled MPEG, RFC 2343, May 1998

[RFC2475] S. Blake et al., An Architecture for Differentiated Services, RFC2475, December 1998

[RFC2581] W. Stevens, M. Allman, V. Paxson, TCP Congestion Control, RFC 2581, April 1999

[RFC2678] J. Mahdavi, V. Paxson, IPPM Metrics for Measuring Connectivity, RFC 2678, September 1999

[RFC2679] G. Almes, S. Kalidindi, and M. Zekauskas, A One-way Delay Metric for IPPM, RFC 2679, September 1999

[RFC2680] G. Almes, S. Kalidindi, and M. Zekauskas, A One-way Packet Loss Metric for IPPM, RFC 2680, September 1999

[RFC2681] G. Almes, S. Kalidindi, M. Zekauskas, A Round-trip Delay Metric for IPPM, September 1999

[RFC2733] J. Rosenberg et al., An RTP Payload Format for Generic Forward Error Correction, RFC 2733, December 1999

[RFC3148] M. Mathis, M. Allman, A Framework for Defining Empirical Bulk Transfer Capacity Metrics, RFC3148, July 2001

[RFC3260] D. Grossman, New Terminology and Clarifications for Diffserv, RFC3260, April 2002

[RFC3261] J. Rosenberg et al., SIP: Session Initiation Protocol, RFC 3261, June 2002

[RFC3357] R. Koodli, R. Ravikanth, One-way Loss Pattern Sample Metrics, RFC3357, August 2002

[RFC3390] M. Allman, S. Floyd, C. Partridge, Increasing TCP's Initial Window, RFC 3390, October 2002

[RFC3393] C. Demichelis, P. Chimento, IP Packet Delay Variation Metric for IP Performance Metrics (IPPM), RFC3393, November 2002

[RFC3550] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 3550, July 2003

[RFC3640] J. van der Meer et al., RTP Payload Format for Transport of MPEG-4 Elementary Streams, November 2003

[RFC3649] S. Floyd, HighSpeed TCP for Large Congestion Windows, RFC 3649, December 2003

[RFC3782] S. Floyd, T. Henderson, A. Gurtov, The NewReno Modification to TCP's Fast Recovery Algorithm, RFC3782, April 2004

[RFC3828] L–A. Larzon et al., The Lightweight User Datagram Protocol (UDP-Lite), RFC 3828, July 2004

[RFC4090] P. Pan, Ed., G. Swallow Ed., A. Atlas Ed., Fast Reroute Extensions to RSVP-TE for LSP Tunnels, RFC 4090, May 2005

[RFC4364] E. Rosen, Y. Rekhter, BGP/MPLS IP Virtual Private Networks (VPNs), RFC 4364, February 2006

[RFC4584] J. Ott et al., Extended RTP profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF), RFC 4584, July 2006

[RFC4588] J. Rey et al., RTP Retransmission Payload Format, RFC 4588, July 2006

[RFC4737] A. Morton et al., Packet Re-ordering Metric for IPPM. RFC 4737, November 2006

[ROSENBERG] J. Rosenberg. G.729 error recovery for Internet Telephony, Project report, Columbia University, 1997

[SAMVIQ] Franc Kozamernik, Paola Sunna, Emmanuel Wyckens and Dag Inge Pettersen, SAMVIQ – A New EBU Methodology For Video Quality Evaluations In Multimedia, Presented at IBC 2004, Amsterdam, September 2004. Available on-line at: http://www.broadcastpapers.com/ibc2004/ibc04EBUSamviq01.htm

[SHELDEN] Nathan Sheldon, Eric Girard, Seth Borg, Mark Claypool, Emmanuel Agu, The effect of latency on user performance in Warcraft III, *Proceedings of the 2nd workshop on Network and system support for games*, pp. 3–14, May 22–23, 2003, Redwood City, California

[SINGH] A. Singh, A. Konrad, and A. Joseph, Performance evaluation of UDP lite for cellular video, in *Proceedings of NOSSDAV*, June 2001.

[STEINMETZ] R. Steinmetz, Human Perception of Jitter and Media Synchronization, *IEEE Journal Selected Areas in Comm.*, vol. 14, pp. 61–72, January 1996

[STEVENS] W. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, Addison-Wesley, 1994

[VC-1] SMPTE 421M-2006, Television – VC-1 Compressed Video Bitstream Format and Decoding Process

Notes

1. Packet header overhead is calculated at 40 bytes comprising: 12 bytes due to RTP header, 8 bytes due to UDP header and 20 bytes due to IP header.
2. The letters in square brackets refer to the components of delay described in Figure 1.5.
3. [ROSENBURG] shows the impact on the Mean Opinion Score (MOS) that loss of consecutive packets has when using a G.729 codec.
4. It is noted that the delay due to the FEC processing operation for a matrix size of m * n reduces in absolute terms as the rate of the encoded MPEG stream increases, because the time to transmit m * n packets reduces accordingly.
5. This paper can be difficult to get hold of, but this advice with respect to response times is also available in [NIELSEN].
6. The nature of the networking industry and community means that some of the sources referred to in this book exist only on the World Wide Web. All Universal Resource Locators (URLs) have been checked and were correct at the time of going to press, but their longevity cannot be guaranteed.