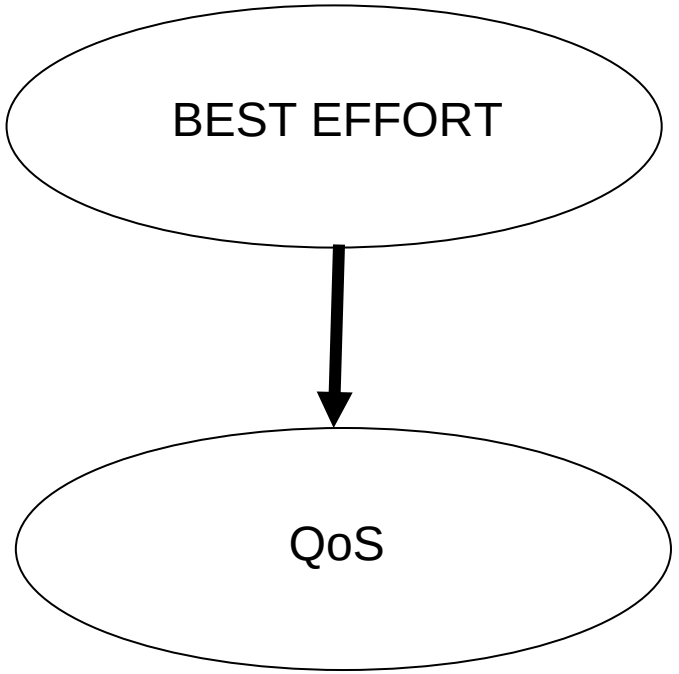
Quality of service in IP networks

- Currently, most Internet services are based on the classic Best-Effort paradigm
- The Best-Effort service does not provide any guarantee on bandwidth, end-to-end delay, packet loss
- The Best-Effort service is very simple and its simplicity has been the most important factor which has determined its worldwide success
- However, the demand of quality associated to the transport of data and media is growing and we are witnessing, in standardizing bodies, in the industry and in the academia, to a significant effort towards a QoS-enabled IP network



Quality of service (QoS)

- Traditionally, QoS is defined in terms of availability, i.e., the percentage time in which the reference system is available and working
- Service Level Agreements are defined as a function of percentage availability, for example 99.975% and of the time required to identify and repair a fault in the connection/equipment
- Usually, a Service Level Agreement is implemented as follows:
  - The system's perimeter to which the SLA applies is defined
  - The availability levels are defined
  - The procedure to measure availability is also defined
  - When the system is in production, its availability is measured
  - If the measured availability is below the minimum threshold, penalties can be applied (they must also be defined in the contract)

Advanced Service Level Agreements

- Real time applications need more advanced SLAs then the simple availability measure
- For example, telephony has strict requirements on delay (related to packet delay and codec delays in the case of VoIP services) and signal quality, related also to packet loss in VoIP services
- More in general, multimedia applications are sensitive to both delay and packet loss
- Advanced SLAs should include packet delay and loss in addition to availability

Real-time and elastic applications

- The quality of some types of applications depends on delay and/or packet loss
- Other applications are robust against delay and/or loss
- For *real-time* applications a packet arriving after a given delay threshold is useless (see for example VoIP)
- Moreover, for real-time applications the retransmission of lost packet is as well useless, as the retransmitted packet is likely to reach its destination after the maximum allowed delay
- Other applications, such as FTP and more in general (even if not always) data transfer, are more robust against delay and they are referred to as *elastic*
- Elastic applications are more robust than real-time applications against packet loss, as they admit retransmission, usually accomplished in an end-to-end fashion through the TCP mechanisms

Real time applications: playback applications (I)

- An important class of real-time applications are *playback* applications (see IETF RFC 1633)
- In a playback application, the source takes some signal, packetizes it, and then transmits the packets over the network
- The network inevitably introduces some variation in the delay of the delivered packets
- The receiver depacketizes the data and then attempts to faithfully play back the signal
- This is done by buffering the incoming data and then replaying the signal at some fixed offset delay from the original departure time
- The term playback point refers to the point in time which is offset from the original departure time by this fixed delay
- Any data that arrives before its associated playback point can be used to reconstruct the signal; data arriving after the playback point is essentially useless in reconstructing the real-time signal

Real time applications: playback applications (II)

- In order to choose a reasonable value for the offset delay, an application needs some "a priori" characterization of the maximum delay its packets will experience
- This "a priori" characterization could either be provided by the network in a quantitative service commitment to a delay bound, or through the observation of the delays experienced by the previously arrived packets
- The performance of a playback application is measured along two dimensions:
  - latency and
  - Fidelity
- Some playback applications, in particular those that involve interaction between the two ends of a connection such as a phone call, are rather sensitive to the latency; other playback applications, such as transmitting a movie or lecture, are not
- Similarly, applications exhibit a wide range of sensitivity to loss of fidelity

Real time applications: playback applications (III)

- There exist two dichotomous classes:
  - *intolerant applications*, which require an absolutely faithful playback (see for example circuit emulation),
  - and *tolerant applications*, which can tolerate some loss of fidelity
- The vast bulk of audio and video applications will be tolerant, but there will be other applications, such as circuit emulation, that are intolerant
- Delay can affect the performance of playback applications in two ways
  - First, the value of the offset delay, which is determined by predictions about the future packet delays, determines the latency of the application
  - Second, the delays of individual packets can decrease the fidelity of the playback by exceeding the offset delay
    - the application then can either change the offset delay in order to play back late packets (which introduces distortion) or
    - merely discard late packets (which creates an incomplete signal)
  - The two different ways of coping with late packets offer a choice between an incomplete signal and a distorted one, and the optimal choice will depend on the details of the application, but the important point is that late packets necessarily decrease fidelity

Real time applications: playback applications (IV)

- Intolerant applications must use a fixed offset delay, since any variation in the offset delay will introduce some distortion in the playback
- For a given distribution of packet delays, this fixed offset delay must be larger than the absolute maximum delay, to avoid the possibility of late packets
- Such an application can only set its offset delay appropriately if it is given a perfectly reliable upper bound on the maximum delay of each packet
- We call a service characterized by a perfectly reliable upper bound on delay *guaranteed service*, and this service is proposed in IETF standards as the appropriate service model for intolerant playback applications
- In contrast, tolerant applications need not set their offset delay greater than the absolute maximum delay, since they can tolerate some late packets

Real time applications: playback applications (V)

- For tolerant applications it can be applied a service model called *predictive service* which supplies a fairly reliable, but not perfectly reliable, delay bound
- This bound, in contrast to the bound in the guaranteed service, is not based on worst case assumptions on the behavior of other flows
- Instead, this bound might be computed with properly conservative (possibly statistical) predictions about the behavior of other flows
- If the network turns out to be wrong and the bound is violated, the application's performance will perhaps suffer, but the users are willing to tolerate such interruptions in service in return for the presumed lower cost of the service

Real time applications: playback applications (VI)

- It is clear that given a choice, with all other things being equal, an application would perform no worse with absolutely reliable bounds than with fairly reliable bounds
- Why, then, do we offer predictive service?
- The key consideration here is efficiency; when one relaxes the service requirements from perfectly to fairly reliable bounds, this increases the level of network utilization that can be sustained, and thus the price of the predictive service will presumably be lower than that of guaranteed service
- The predictive service class is motivated by the conjecture that the performance penalty will be small for tolerant applications but the overall efficiency gain will be quite large

Elastic applications (I)

- While real-time applications do not wait for late data to arrive, elastic applications will always wait for data to arrive
- It is not that these applications are insensitive to delay; to the contrary, significantly increasing the delay of a packet will often harm the application's performance
- Rather, the key point is that the application typically uses the arriving data immediately, rather than buffering it for some later time, and will always choose to wait for the incoming data rather than proceed without it
- Because arriving data can be used immediately, these applications do not require any a priori characterization of the service in order for the application to function
- Generally speaking, it is likely that for a given distribution of packet delays, the perceived performance of elastic applications will depend more on the average delay than on the tail of the delay distribution
- One can think of several categories of such elastic applications: interactive burst (Telnet, NFS), interactive bulk transfer (FTP), and asynchronous bulk transfer (electronic mail, FAX)
- The delay requirements of these elastic applications vary from rather demanding for interactive burst applications to rather lax for asynchronous bulk transfer, with interactive bulk transfer being intermediate between them

Elastic applications (II)

- An appropriate service model for elastic applications is to provide "as-soon-as-possible", or ASAP service
- For compatibility with historical usage, the term best-effort service is used as a synonym of ASAP service
- By offering several classes of best-effort it is possible to serve appropriately the different delay sensitivities of different elastic applications
- The multi-tier best-effort service model allows interactive burst applications to have lower delays than interactive bulk applications, which in turn would have lower delays than asynchronous bulk applications
- In contrast to the real-time service models, applications using this service are not usually (but not necessarily) subject to admission control

On the application taxonomy

- The taxonomy of applications into tolerant playback, intolerant playback, and elastic is not a precise standard, it is a guideline for the proper classification of applications and the type of related service
- Not all real-time applications are playback applications; for example, one might imagine a visualization application which merely displayed the image encoded in each packet whenever it arrived
- However, non-playback applications can still use either the guaranteed or predictive real-time service model
- Similarly, playback applications cannot be neatly classified as either tolerant or intolerant, but rather fall along a continuum
- Offering both guaranteed and predictive service allows applications to make their own tradeoff between fidelity, latency, and cost

General service models: resource allocation

- The network must allocate its resources among traffic flows to meet quality objectives
- This allocation of resources can be negotiated on
  - an individual flow-by-flow basis or
  - Taking into account traffic aggregates, i.e., individual traffic flows are grouped into service categories, if the have similar QoS requirements
- As far as QoS is concerned, delay is one of the fundamental metrics and also packet loss is important
- For resource allocation the quantity of primary interest in resource-sharing is aggregate bandwidth on individual links
- Thus, this component of the service model, called *link-sharing*, addresses the question of how to share the aggregate bandwidth of a link among various collective entities
- Link sharing can be carried out in various fashions
  - Multi-entity
  - Multi-protocol
  - Multi-service

Multi-entity link-sharing

- In multi-entity link-sharing a link may be purchased and used jointly by several client organizations
- They may wish to insure that under overload the link is shared in a controlled way, for example in proportion to the capital investment of each entity
- At the same time, they might wish that when the link is underloaded, any one of the entities could utilize all the idle bandwidth

Multi-protocol link-sharing

- Multi-protocol link-sharing aims at preventing that one protocol family (DECnet, IP, IPX, OSI, SNA, etc.) overloads the link excluding the other families
- This is important because different families may have different methods of detecting and responding to congestion, and some methods may be more "aggressive" than others
- This could lead to a situation in which one protocol backs off more rapidly than another under congestion, and ends up getting no bandwidth
- Explicit control by routers is required
- One might expect that this control should apply only under overload, while permitting an idle link to be used in any proportion

Multi-service link-sharing

- With multi-service sharing, within a protocol family such as IP, an administrator might wish to limit the fraction of bandwidth allocated to various service classes
- For example, an administrator might wish to limit the amount of real-time traffic to some fraction of the link, to avoid preempting elastic traffic such as FTP
- In this way, it is possible to differentiate the service of different protocols and to guarantee the QoS objectives of all protocols

Link-sharing, general considerations

- In the link-sharing service model the aggregate bandwidth is shared according to some specified shares
- For instance, a link could be divided between a number of organizations, each of which would divide the resulting allocation among a number of protocols, each of which would be divided among a number of services
- Thus, it is possible to implement a hierarchical link sharing policy
- Admission control is necessary to ensure that the real-time service commitments can be met
- Similarly, admission control will again be necessary to ensure that the link-sharing commitments can be met
- For each entity, admission control must keep the cumulative guaranteed and predictive traffic from exceeding the assigned link-share

Example: classes of relevant applications

- Telephony service class is best suited for applications that require very low delay variation and are of constant rate, such as IP telephony (VoIP) and circuit emulation over IP applications
- Signaling service class is best suited for peer-to-peer and client-server signaling and control functions using protocols such as SIP, SIP-T, H.323, H.248, and Media Gateway Control Protocol (MGCP)
- Multimedia Conferencing service class is best suited for applications that require very low delay and have the ability to change encoding rate (rate adaptive), such as H.323/V2 and later video conferencing service
- Real-Time Interactive service class is intended for interactive variable rate inelastic applications that require low jitter and loss and very low delay, such as interactive gaming applications that use RTP/UDP streams for game control commands, and video conferencing applications that do not have the ability to change encoding rates or to mark packets with different importance indications
- Multimedia Streaming service class is best suited for variable rate elastic streaming media applications where a human is waiting for output and where the application has the capability to react to packet loss by reducing its transmission rate, such as streaming video and audio and webcast
- Broadcast Video service class is best suited for inelastic streaming media applications that may be of constant or variable rate, requiring low jitter and very low packet loss, such as broadcast TV and live events, video surveillance, and security
- Low-Latency Data service class is best suited for data processing applications where a human is waiting for output, such as web-based ordering or an Enterprise Resource Planning (ERP) application
- High-Throughput Data service class is best suited for store and forward applications such as
- Low-Priority Data service class is intended for packet flows where bandwidth assurance is not required

Examples of quality requirements of a set of relevant applications

Application	Characteristics	Tolerance to loss	Tolerance to delay	Tolerance to jitter
Network control	Mostly inelastic	Low	Low	Yes
Telephony	Inelastic, low rate	Very low	Very low	Very low
Signaling	Shoret packets, delay critical	Low	Low	Yes
Multimedia conferencing	Reacts to loss	Low medium	Very low	Low
Real-time interactive	Inelastic, variable bit rate	Low	Very low	Low
Multimedia streaming	Elastic, variable bit rate	Low medium	Medium	Yes
Broadcast video	Inelastic, non variable bit rate	Very low	Medium	Low
Low latency data	Elastic, variable bit rate	Low	Low medium	Yes
OAM	Both elastic and inelastic	Low	Medium	Yes
High throughput data	Elastic	Low	Medium high	Yes
Standard	A bit of everything	n.a	n.a	n.A
Low-priority data	elastic	High	High	yes

The "infinite bandwidth" paradigm

- The incredibly large carrying capacity of an optical fiber leads some to conclude that in the future bandwidth will be so abundant, ubiquitous, and cheap that there will be no communication delays other than the speed of light, and therefore there will be no need to reserve resources
- However, this is impossible in the short term and unlikely in the medium term
- While raw bandwidth may seem inexpensive, bandwidth provided as a network service is not likely to become so cheap that wasting it will be the most cost-effective design principle
- Even if low-cost bandwidth does eventually become commonly available, it is neither proved nor likely that it will be available "everywhere" in the Internet
- Unless we provide for the possibility of dealing with congested links, then real-time services will simply be precluded in those cases
- This restriction is unacceptable
- Therefore, the usage of bandwidth must be controlled

The role of customers and providers

- The end-to-end guarantee of packet delay and loss is very difficult
- This is why delay-oriented and packet loss-oriented SLAs are not frequent
- However, the demand for added-value transport services is increasing, especially from business and industry customers
- Therefore operators are starting to offer more advanced SLAs (more frequently for important clients) including packet delay and packet loss in the SLA's metrics
- However, the process is difficult and it can be observed that in some cases the operator's and the customer's views are different and that finding a convergence is difficult

Sample delay-oriented Service level Agreement (I)

- This is an example of a real set of SLAs offered by an operator
- This offer is based on the *Olympic Services* framework
- The Gold service is the best, the silver service is the second and the bronze service is the last
- Note that the Best-Effort service is also offered, but no SLA is assigned to this service

SLA (QOS)	Access link speed: 2.048 Mbit/s MAX E2E delay and maximum fraction *p* of packets that can exceed max delay	Access link speed: 4 Mbit/s MAX E2E delay and maximum fraction *p* of packets that can exceed max delay	…
Gold:	*d*=90 ms, *p*=0.001	*d*=80 ms, *p*=0.001	…
Silver:	*d*=120 ms, *p*=0.001	*d*=100 ms, *p*=0.001	…
Bronze:	*d*=250 ms, *p*=0.001	*d*=210 ms, *p*=0.001	…

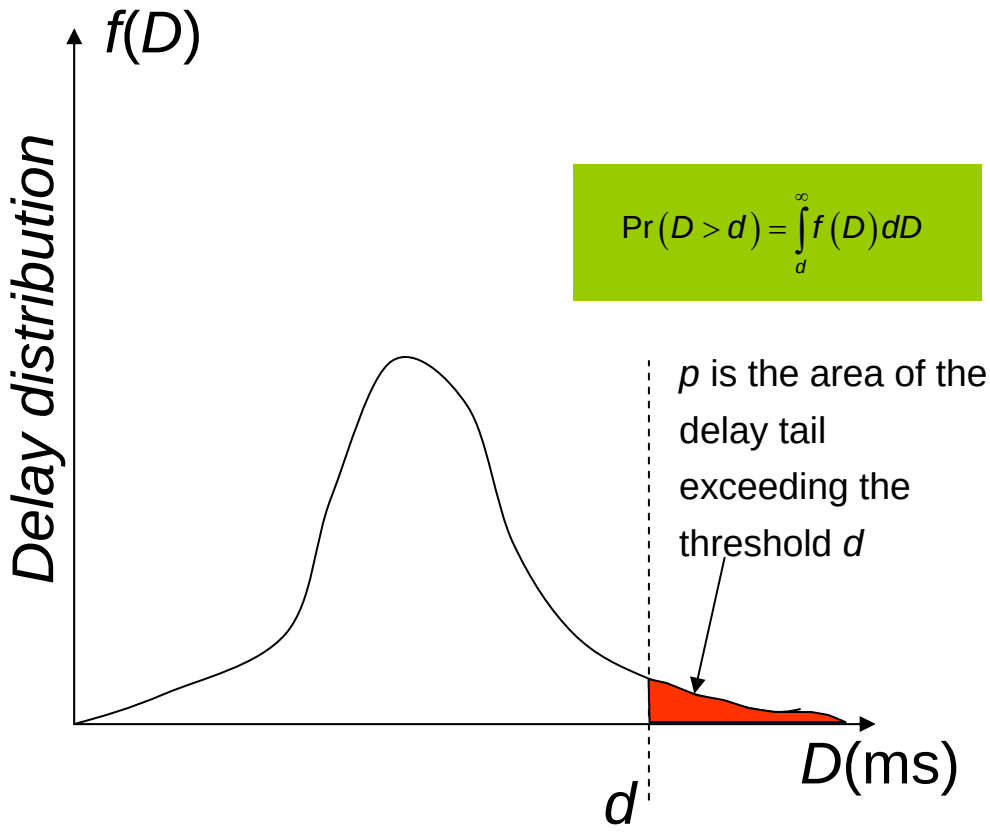
Sample delay-oriented Service level Agreement (II)

- Note tha SLAs are defined *statistically*
- A maximum delay threshold *d* is fixed
- A maximum fraction *p* of packets can be allowed to exceed the delay threshold *d*
- Formally, the SLA is defined as
  - Pr(*D*>*d*)<*p*
  - Where *D* is the actual delay of packets

SLA (QOS)	Access link speed: 2.048 Mbit/s MAX E2E delay and maximum fraction *p* of packets that can exceed max delay	Access link speed: 4 Mbit/s MAX E2E delay and maximum fraction *p* of packets that can exceed max delay	…
Gold:	*d*=90 ms, *p*=0.001	*d*=80 ms, *p*=0.001	…
Silver:	*d*=120 ms, *p*=0.001	*d*=100 ms, *p*=0.001	…
Bronze:	*d*=250 ms, *p*=0.001	*d*=210 ms, *p*=0.001	…

Sample delay-oriented Service level Agreement (III)

- A statistical delay SLA is defined on the basis of the distribution of packet delay, *f*(*D*)
- Given the packet delay distribution, the probability of exceeding a delay threshold *d* is the area under the curve *f*(*D*), from *d* to infinity
- Thus, statistical delay SLAs are based on the concept of the *delay tail*, whose weight (area) should not exceed a preassigned measure, *p*
- A statistical delay SLA can be indicated with the short-hand notation (*d*, *p*)



Sample delay-oriented Service level Agreement (IV)

- A statistical delay SLA could be implemented as follows
  - The (*d*, *p*) SLA is defined
  - The SLA is measured on the working system (for example, the number of packets exceeding the delay threshold should be counted over pressigned time intervals)
  - A summary evaluation of the received QoS should be produced on the basis of the measured data
  - The customer can establish if it has received the contracted QoS
- In real life, it is not always easy to reach an agreement on how QoS is measured

Sample delay-oriented Service level Agreement (V)

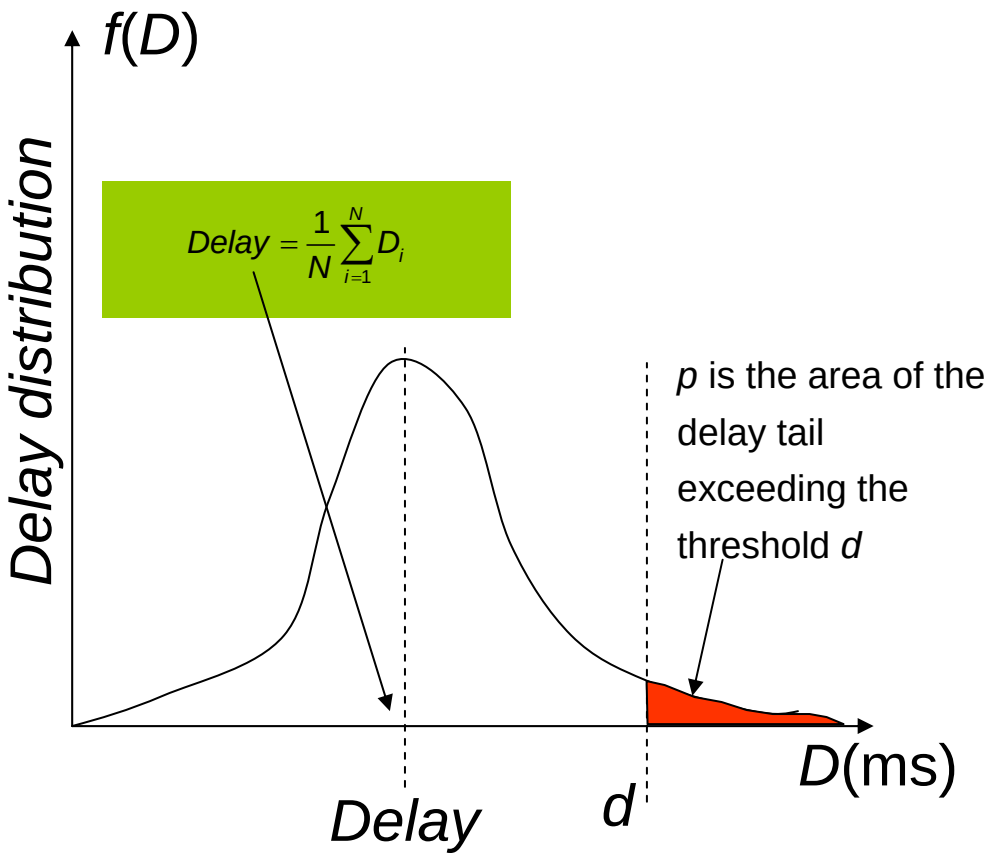
- For example, the provider TELCO proposes to its customer ACME the following method to measure quality
- Given a SLA (d, p):
  - Periodically, every Δt (s), an end-to-end ping packet is sent to measure delay
  - Let us define D_i as the delay of the ith ping packet
  - In a time period $T > \Delta t$, T/ Δt = N ping packets are sent and their delay is measured
  - The following formula to calculate delay is proposed

$$Delay = \frac{1}{N}\sum_{i=1}^{N} D_i$$

- If Delay>d, the SLA has been violated
- *Is there anything strange with this method?*

Sample delay-oriented Service level Agreement (VI)

- Yes, there is something strange!
- *Delay* is actually the average value of end-to-end delay
- It is practically impossible that the average delay is greater than a delay threshold *d*, if *d* is set greater than average delay
- In this way, the result of the measures would be that the SLA is always fulfilled, even in the case in which the fraction of packets exceeding the delay threshold d is greater than *p*
- Note also that the *p* parameter is not considered in the proposed method



Sample delay-oriented Service level Agreement (VII)

- The customer, ACME, complains with the provider TELCO about this way of measuring the SLA
- ACME asks for a better method and TELCO replies with an alternative proposal
  - The previous system of ping packets is maintained
  - The 10% of highest delays is removed
  - Moreover, if a delay measure *Di* occurs when a router (any router) or a link (any link) is overloaded, that measure is removed
- Clearly, also with this system the SLA results as always fulfilled
- In fact, a high delay is necessarily a consequence of a loaded router and/or link and, with this definition, all highest delays are removed

Sample delay-oriented Service level Agreement (VIII)

- How a thorough method for measuring a statistical delay SLA could be defined?
- For example:
- Given a SLA (d, p):
  - Periodically, every Δt (s), an end-to-end ping packet is sent to measure delay
  - Let us define D_i as the delay of the ith ping packet
  - In a time period $T > \Delta t$, T/ $\Delta t = N$ ping packets are sent and their delay is measured
  - Let N_1 be the number of packets exceeding the delay threshold d
  - If $N_1/N > p$, the SLA has been violated

Sample delay-oriented Service level Agreement (IX)

- This experience suggests that guaranteeing end-to-end statistical delay SLAs is difficult
- Provider may want to seek for some protection in the contract with customers
- The correct path towards a thorough QoS guarantee is to acquire the knowhow on how to manage bandwidth in order to assign to each service class the amount of resources necessary and sufficient to obtain the required QoS