- Introduction to QoS -

Obstacles to Network Communication

Modern networks support traffic beyond the traditional data types, such as email, file sharing, or web traffic. Increasingly, data networks share a common medium with more sensitive forms of traffic, like *voice* and *video.*

These sensitive traffic types often require *guaranteed* or *regulated* service, as such traffic is more susceptible to the various obstacles of network communication, including:

Lack of Bandwidth – Describes the simple lack of sufficient throughput, which can severely impact sensitive traffic. Increasing bandwidth is generally considered the *best* method of improving network communication, though often expensive and time-consuming.

Bandwidth is generally measured in **bits-per-second (bps)**, and can be offered at a fixed-rate (as Ethernet usually is), or at a variable-rate (as Frame-Relay often is). Various mechanisms, such as **compression**, can be used to pseudo-increase the capacity of a link.

Delay – Defines the latency that occurs when traffic is sent end-to-end across a network. Delay will occur at various points on a network, and will be discussed in greater detail shortly.

Jitter – Describes the fragmentation that occurs when traffic arrives at irregular times or in the wrong order. Jitter is thus a *varying* amount of delay. Voice communication is *especially* susceptible to jitter. Jitter can be somewhat mitigated using a **de-jitter buffer**.

Data Loss – Defines the *packet loss* that occurs due to link congestion. A full queue will drop newly-arriving packets - an effect known as **tail drop**.

All of above factors adversely affect network communication. Voice over IP (VoIP) traffic, for example, begins to degrade when delay is higher than **150 ms**, and when data loss is greater than **1%**.

Quality of Service (QoS) tools have been developed as an alternative to merely increasing bandwidth. These QoS mechanisms are designed to provide specific applications with guaranteed or consistent service in the absence of optimal bandwidth conditions.

Types of Delay

Delay can occur at many points on a network. Collectively, this is known as **end-to-end delay**. The various *types* of delay include:

- **Serialization Delay –** refers to the time necessary for an interface to encode bits of data onto a physical medium. Calculating serialization delay can be accomplished using a simple formula:

  $$\frac{\textbf{\# of bits}}{\textbf{bits per second (bps)}}$$

  Thus, the serialization delay to encode 128,000 bits on a 64,000 bps link would be 2 seconds.

- **Propagation Delay** – refers to the time necessary for a single bit to travel end-to-end on a physical wire. For the incredibly anal geeks, the rough formula to estimate propagation delay on a copper wire:

  $$\frac{\textbf{Length of the Physical Wire (in meters)}}{\mathbf{2.1 \times 10^{8}}\ \textbf{meters/second}}$$

- **Forwarding** (or **Processing) Delay –** refers to the time necessary for a router or switch to move a packet between an *ingress* (input) queue and an *egress* (output) queue. Forwarding delay is affected by a variety of factors, such as the routing or switching method used, the speed of the device's CPU, or the size of the routing table.

- **Queuing Delay** – refers to the time spent in an egress queue, waiting for previously-queued packets to be serialized onto the wire. Queues that are too small can become congested, and start dropping newly arriving packets (**tail drop).** This forces a higher-layer protocol (such as TCP) to resend data**.** Queues that are too large can actually queue too many packets, causing long queuing delays.

- **Network (Provider) Delay –** refers to the time spent in a WAN provider's cloud. Network delay can be very difficult to quantify, as it is often impossible to determine the structure of the cloud.

- **Shaping Delay –** refers to the delay initiated by shaping mechanisms intended to slow down traffic to prevent dropped packet due to congestion.

QoS Methodologies

There are three key methodologies for implementing QoS:

- **Best-Effort**
- **Integrated Services (IntServ)**
- **Differentiated Services (DiffServ)**

Best-Effort QoS is essentially *no* QoS. Traffic is routed on a first-come, first-served basis. Sensitive traffic is treated no differently than normal traffic. Best-Effort is the default behavior of routers and switches, and as such is easy to implement and very scalable. The Internet forwards traffic on a Best-Effort basis.

Integrated Services (IntServ) QoS is also known as *end-to-end* or *hard* QoS. IntServ QoS requires an application to *signal* that it requires a specific level of service. An **Admission Control** protocol responds to this request by allocating or reserving resources end-to-end for the application. If resources *cannot* be allocated for a particular request, then it is denied.

Every device end-to-end must support the IntServ QoS protocol(s). IntServ QoS is not considered a scalable solution for two reasons:

- There is only a finite amount of bandwidth available to *reserved.*
- IntServ QoS protocols add significant overhead on devices end-to-end, as each traffic flow must be statefully maintained.

The **Resource Reservation Protocol (RSVP)** is an example IntServ QoS protocol.

Differentiated Services (DiffServ) QoS was designed to be a scalable QoS solution. Traffic types are organized into specific **classes,** and then **marked** to identify their classification. **Policies** are then created on a *per-hop basis* to provide a specific level of service, depending on the traffic's classification.

DiffServ QoS is popular because of its scalability and flexibility in enterprise environments. However, DiffServ QoS is considered *soft* QoS, as it does not absolutely guarantee service, like IntServ QoS. DiffServ QoS does not employ signaling, and does not enforce end-to-end reservations.

QoS Tools

Various tools have been developed to enforce QoS. Many of these tools are used in tandem as part of a complete QoS policy:

- **Classification and Marking**
- **Queuing**
- **Queue Congestion Avoidance**

Classification is a method of identifying and then organizing traffic based on service requirements. This traffic is then **marked** or *tagged* based on its classification, so that the traffic can be differentiated. Classification and marking are covered in great detail in another guide.

Queuing mechanisms are used to service *higher* priority traffic before *lower* priority traffic, based on classification. A variety of queuing methods are available:

- First-In First-Out (FIFO)
- Priority Queuing (PQ)
- Custom Queuing (CQ)
- Weighted Fair Queuing (WFQ)
- Class-Based Weighted Fair Queuing (CBWFQ)
- Low-Latency Queuing (LLQ)

Each will be covered in detail in a separate guide.

Queue Congestion Avoidance mechanisms are used to regulate queue usage so that saturation (and thus, tail drop) does not occur. Random Early Detection (RED) and Weighted RED (WRED) are two methods of congestion avoidance, and are both covered in a separate guide.

Configuring QoS on IOS Devices

There are four basic methods of implementing QoS on Cisco IOS devices:

- **Legacy QoS CLI**
- **Modular QoS CLI**
- **AutoQoS**
- **Security Device Manager (SDM) QoS Wizard**

Legacy QoS CLI is a limited and deprecated method of implementing QoS via the IOS command-line. Legacy CLI combined the *classification* of traffic with the *enforcement* of QoS policies. All configuration occurs on a per-interface basis.

Modular QoS CLI (MQC) is an improved command-line implementation of QoS. MQC is considered *modular* because it separates classification (using **class-maps** to match traffic) from policy configuration (using **policy-maps** to apply a specific level of service per classification). Policy-maps are then applied to an interface using a **service-policy.**

AutoQoS is an automated method of generating QoS configurations on IOS devices. AutoQoS, originally developed for VoIP traffic, can run a *discovery* process to analyze and classify a variety of traffic types. AutoQoS can then create QoS policies based on those classifications. Afterwards, MQC can be used to fine-tune AutoQoS's generated configuration.

The **Cisco Security Device Manager (SDM)** is a web-based management GUI for Cisco IOS devices. The SDM **QoS Wizard** provides a graphical method of configuring and monitoring QoS. The Wizard separates traffic into three categories:

- Real-Time – for VoIP and signaling traffic.
- Business-Critical – for transactional, network management, and routing traffic.
- Best Effort – for all other traffic.

A percentage of the interface bandwidth can then be allocated for each traffic category.

MQC and AutoQoS will be covered in greater detail in separate guides.