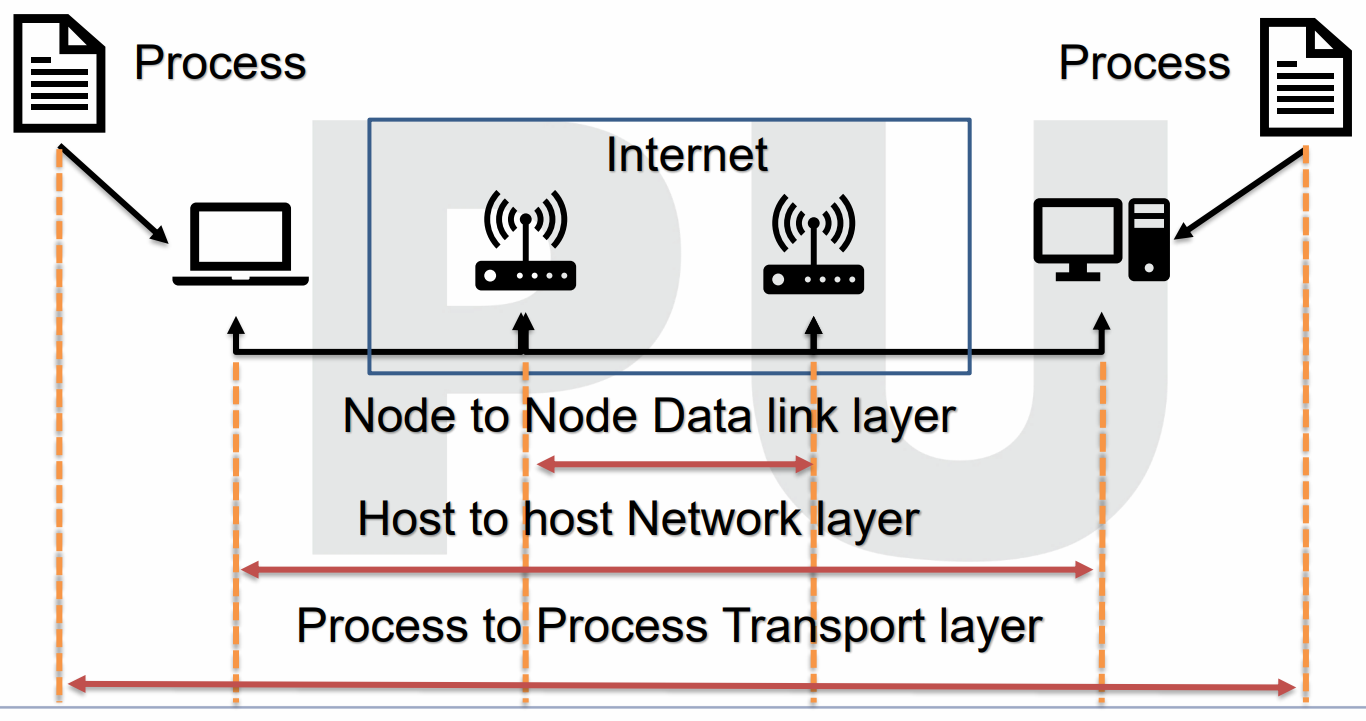
**TRANSPORT LAYER**

**Transport Layer Services & Protocol**

* It is responsible for **reliable transportation** of data.
* ***Transport layer*** is **more** **abstracted** (high-level) than ***network layer***.
* **Sender’s side** **–** Breaks application messages into small pieces & passes them to ***network layer***.
* **Receiver’s side –** Joins the pieces into message again & passes to ***session layer***.
* For example: **TCP**, **UDP** etc.

**Process to Process Communication**

* ***Data link layer*** is responsible for transmitting frames **between two nodes**.
* ***Network layer*** is responsible for delivering ***datagrams*** **between two hosts**.
* **Datagram:** Head, tail & main data as a whole.
* Each process in ***application layer*** has to communicate with another process continuously.
* ***Transport layer*** breaks these processes down into **small pieces**.



**Client-Server Paradigm**

* A **local host** (***client***) uses a ***server*** to communicate resources.

Requirements for communication:-

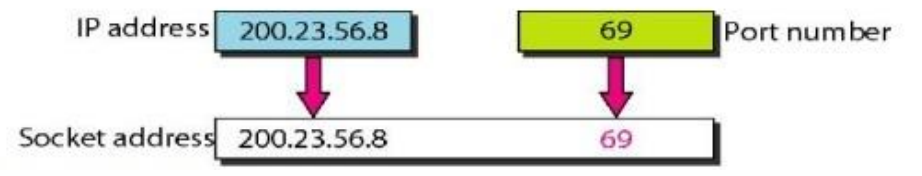
* Local host
* Local process
* Remote host
* Remote process

**Addressing**

* We require **MAC address** for sending data at ***data link layer***.
* But it is required **only when** the communication is **not point-to-point**.
* **No point-to-point** means that data has to traverse across **multiple nodes** while communication.
* Other than **MAC address**, the **source address** is also required.
* **Source address:** Contains information that **helps in traversing** of data to **next node**.
* In addressing, **IP address** is used for identifying the **destination host uniquely** among other hosts connected to the same network.
* While **MAC address** also identifies all hosts uniquely, **IP address** also **ensures** that they are connected to common network.
* **Port number:** Helps identifying all **applications uniquely** on same network.

**Socket Addresses**

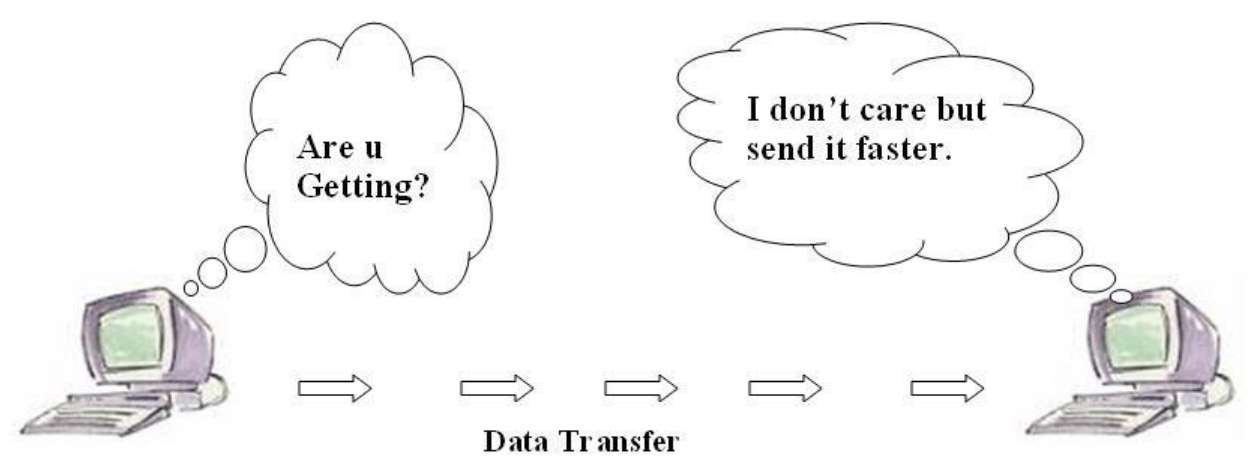
* **Socket address:** Combination of ***IP address*** & ***port number***.



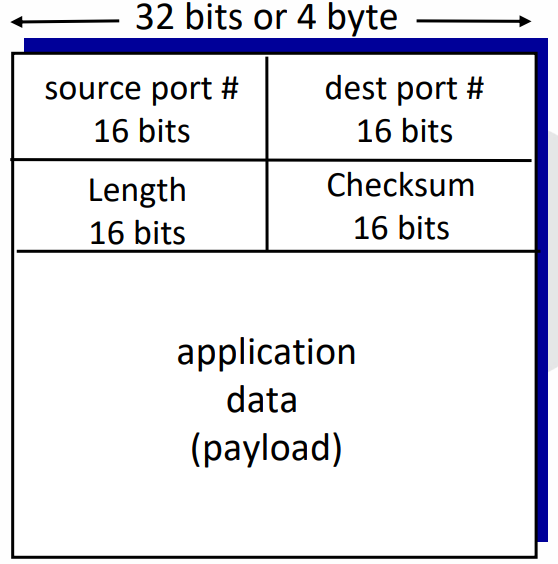
* And ***socket address*** identifies **individual client** process uniquely.
* **Address socket address:** Uniquely defines a ***server process***.

**User Datagram Protocol (UDP)**

* It is a ***transport layer*** protocol.
* Application’s process passes the data to ***network layer*** after **compression**.
* This **compression** is done using ***multiplexer*** on **sender’s side**.
* Then ***extraction*** is done using **demultiplexer** on **receiver’s side**.
* In UDP, **neither** receiver sends acknowledgement **nor** sender waits for it.



**UDP Segment Header**



* Connection is **not** established first.
* Small header size.
* **No congestion control**, UDPs blast away as fast as possible.
* **Source port:** **16-bit** long port number of **source**.
* **Destination port:** **16-bit** long port number of **destination**.

**Checksum**

* **16-bit** long address.
* Used for **detecting errors** in transmitted segment.
* UDP contains **header segment** & **checksum value**.
* **Checksum value:** Sum of 1’s complement of **all contents** of segment.
* **Sender** calculates **checksum value** & assigns it to **checksum value field** in UDP.
* **Receiver** recalculates **checksum value** & **cross checks** it with the sender’s checksum.
* This cross checking is done through **XORing** with where error free value must be **0**.
* Cross checking them tells if there is any **error or not**.

**Applications of UDP**

* **Domain name services (DNS):** Converts **human readable** domain names into **IP addresses**.
* **Simple network management protocol (SNMP):** Used for **monitoring** and **managing** devices connected to network.
* **Trivial file transfer protocol (TFTP):** Protocol used for **transferring files** between client and server.
* **Routing information protocol (RIP):** Used for **dynamically communicating** about routing information.
* **Kerberos:** A protocol used for **aiding** client-server applications for **authentication** of users & also provides various other security functionalities.

**Transmission Control Protocol (TCP) & Internet Protocol (IP)**

* **Collection of protocols** that connect computers to internet.
* It defines **most** of internet structures & functions; including **packaging**, **addressing**, **sending** & **collecting** etc.

Characteristics of TCP/IP:-

* **Share data transfer:** The ability to build **large networks** & make **reliable** transfer of data.
* **Reliable**, as it retrieves lost & misplaced data.
* Uses **multiplexing**.

Applications:-

* **Simple mail transfer protocol (SMTP):** Used for **transferring email** from one email address to another.
* **File transfer protocol (FTP):** Used for sending **large sized** files.
* **Dynamic host configure protocols (DHCP):** Assigns IP address.
* **Telnet:** Enables text communication via a **terminal** application.
* **Hyper text transfer protocol (HTTP):** Used for transferring **web pages**.
* **Domain name service (DNS)**
* **Simple network time protocol (SNTP):** Enables network devices to **access time** of day.

**Stream Control Transfer Protocol (SCTP)**

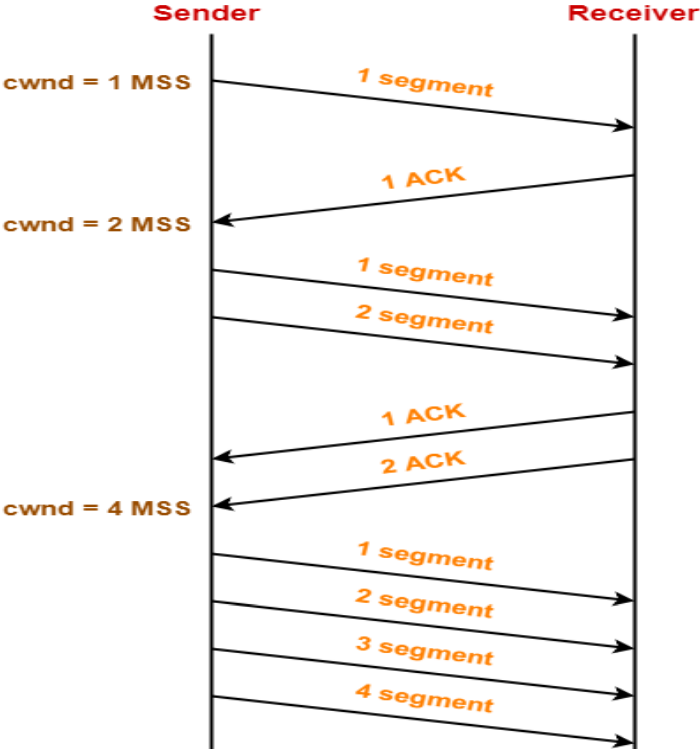
* A **safe transport protocol**; running parallel to unsafe one, like **IP**.
* Using **checksums** & other error checking methods, it ensures **error free** & **non-duplicate** message transfers.
* It has inherited many features of **TCP** including **congestion management** & **lost packet detection** etc.

**Congestion Control**

* **Congestion** happens when a **huge amount** of data is fed to a device, the device **not** being able to process it.
* **Windows mechanism:** Congestion management method used by **TCP**.
* In **windows mechanism**, a time limit is set on the sender to send data.
* This avoids a device from being fed **excessive** data.
* Steps TCP takes for congestion control:
  + Slow start
  + Congestion avoidance
  + Congestion detection

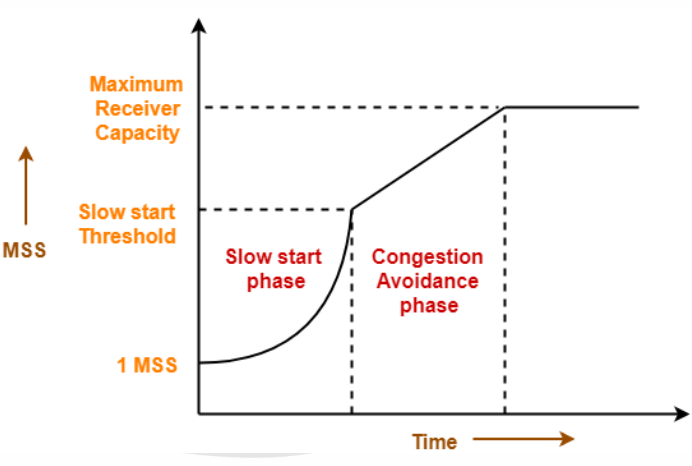
Slow start:-

* A **congestion window size** is given at first, measured in unit called **MSS**.
* At starting it is set to MSS.
* Then when the sender receives an acknowledgement, it expands the window **exponentially** by increasing **2’s power** (2 MSS here).
* Then 4 MSS, then 8 MSS and so on.



**Congestion Avoidance**

* After the phase of **slow start** ends, phase of **congestion avoidance** starts.
* Here, the **congestion window** rises **linearly**.
* For each acknowledgement, **congestion window** size is raised by **1**.



**Congestion Detection**

**Case 1:** Detection on time out:-

* **Time outs** before receiving acknowledgement.
* Chances are that the congestion is **serious**.
* Rate of congestion threshold is **exponentially decreased** until it becomes **1 MSS**.

**Case 2:** Detection on duplicate acknowledgements:-

* Probability of message **not** being reached is **less**.
* Still there might be chances for it.
* Reduce the **duration** of congestion by **half** & window congestion too.

**Quality of Service (QoS)**

* It’s a set of **traffic control** mechanisms that act on the basis of **network performance** or the **network requirements** of the operator.
* It takes calculated steps to **enhance** network performance.
* For ***Qos***, low performance generally means **high packet delay** & **high number of** **packet loss**.

Applications:-

* Video & audio conferencing/streaming.
* Hard real-time database management.

What QoS improves:-

* Delay
* Delay variation (jitter/flickering)
* Throughput
* Error rate

Types of QoS solutions:-

* Stateless solutions
* Stateful solutions

**Stateless Solutions**

* **Doesn’t** provide a very efficient control over traffic.
* But is highly **scalable** and **resilient** to possible internal damages.
* However, its **unreliable** of results.

**Stateful Solutions**

* Provides **high control** over traffic.
* **Not** much **flexible** or **resilient** for internal damages.
* Highly **reliable** of results.
* It also maintains the smoothness in **flow of information**.

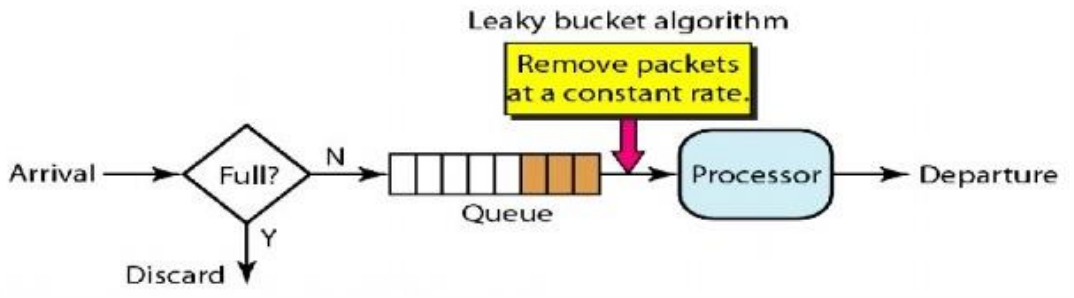
**QoS Improvement Techniques**

* Any method that can improve **service efficiency**.
* ***Traffic shaping*** is one such method.
* **Traffic shaping:** Regulating **congestion volume** & **rate of flow** of information.

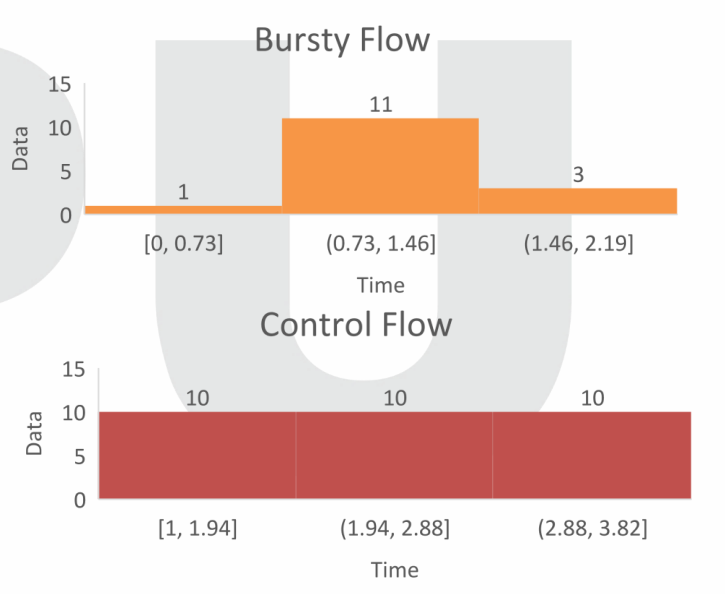
Proper traffic forming method:-

* Leaky bucket
* Token bucket

**Leaky Bucket**

****

* This method is named this because it behaves similar to how **buckets** behave during **leakage**.
* A bucket with a hole spills water at the **same rate**, whether more water is added to it or not & at whatever rate.
* Similarly, in this method the flow of data **remains smooth** & **undisturbed** by the number and rate of input being fed.
* High number of inputs are processed & then **pushed** in a stable flow.
* It basically converts a **bursty flow** into **controlled flow**.



**Token Bucket**

* **Stricter** than leaky bucket.
* If a host stays **idle** for too long, then resources in its bucket (**queue**) are **emptied**.
* Thus, it becomes **null**.
* But this method enables a host to **store tokens** for future use.
* **Tokens:** **N** **credits** provided per system clock’s tick.
* These tokens are used by the **host** when their chance comes for **data transmission**.
* One must have **enough tokens** for being able to send data.
* If not, then it has to **wait** until it has enough tokens.
* This token system **regulates the traffic** by fixing **limited data flow** across networks.

