What Transport Service Does an App Need?

Transport Service Needs

Different applications require different services from the transport layer.

- **Data Integrity:**
  - Some apps (e.g., file transfer, web transactions) require 100% reliable delivery.
  - Other apps (e.g., audio, video) can tolerate some packet loss.
- **Throughput:**
  - Some apps (e.g., multimedia streaming) require a minimum throughput to be effective.
  - Others (e.g., email, file download) are **elastic**, meaning they adapt to whatever throughput is available.
- **Timing:**
  - Real-time apps (e.g., Internet telephony, interactive games) require low delay.
  - Non-interactive apps (e.g., email, file transfer) are insensitive to delay.
- **Security:**
  - Applications may require encryption, integrity checking, and authentication.

Analogy

Think of transport services like a **delivery service**:

- **Data integrity** = making sure no items are missing or damaged.
- **Throughput** = size of the delivery truck (how much can be carried per trip).
- **Timing** = how fast the delivery arrives (express vs. normal shipping).
- **Security** = whether the package is sealed and tamper-proof.

Transport Requirements of Common Applications

Application	Data Loss	Throughput	Time Sensitive?
File transfer / Download	No loss	Elastic	No
E-mail	No loss	Elastic	No
Web documents	No loss	Elastic	No
Real-time audio/video	Loss-tolerant	5Kbps–5Mbps	Yes (10–100 ms)
Streaming audio/video	Loss-tolerant	5Kbps–5Mbps	Yes (seconds acceptable)
Interactive games	Loss-tolerant	Kbps+	Yes (10s of ms)
Text messaging	No loss	Elastic	Sometimes

Analogy

Applications are like **different passengers at an airport**:

- Some (file transfer, email) don't mind waiting—they only care about **arriving safely**.
- Some (video calls, games) want **fast boarding with minimal delay**.
- Some (streaming) can buffer a bit but still need a steady flow.

Internet Transport Protocols

TCP Service

- **Reliable transport** between sending and receiving processes.
- **Flow control:** prevents sender from overwhelming receiver.
- **Congestion control:** reduces sending rate if the network is overloaded.
- **Connection-oriented:** setup required between client and server.
- **Limitations:** provides no timing guarantees, no throughput guarantee, no built-in security.

UDP Service

- **Unreliable data transfer** (packets may be lost or out-of-order).
- **No extras:** no reliability, no flow control, no congestion control, no timing, no security, no connection setup.

- Often used for delay-sensitive apps (e.g., games, voice, video).

Analogy

- **TCP** = registered mail: reliable, tracked, must be signed for, but slower.
- **UDP** = postcards: fast, lightweight, but no guarantee of delivery.

Applications and Transport Protocols

Application	Application-Layer Protocol	Transport Protocol
File transfer / Download	FTP [RFC 959]	TCP
E-mail	SMTP [RFC 5321]	TCP
Web documents	HTTP [RFC 7230, 9110]	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary	TCP or UDP
Streaming audio/video	HTTP, DASH	TCP
Interactive games	Proprietary (FPS, MMO, etc.)	UDP or TCP

Securing TCP

Vanilla TCP/UDP Sockets

- No encryption by default.
- Cleartext passwords and data traverse the Internet unprotected (!).

Transport Layer Security (TLS)

- Provides **encrypted TCP connections**.
- Ensures **data integrity**.
- Provides **end-point authentication**.
- Implemented in the application layer (apps use TLS libraries, which in turn use TCP).
- Cleartext entered into a TLS-enabled socket is transmitted across the Internet in encrypted form.

Analogy

TLS is like sending mail in a **locked box** with a secret key:

- Without TLS: anyone handling the mail can open and read it.
- With TLS: only the intended recipient has the key to unlock it.