Preview

- Introduction to Client-Server Model
  - Motivation of Client-Server Model
  - Terminologies and Concepts in Client-Server Model
  - Connectionless vs. Connection-Oriented
  - Stateless vs. Stateful Server
  - Identify a client by Stateful Server
  - Server as a Client

Motivation of Client-Server Model

- The fundamental motivation for the client-server paradigm arises from the problem **rendezvous**.
  - A sender will send a message when it is ready to send.
  - But receiver does not know when sender send a message to the receiver.
- The client-server model solves the rendezvous problem in a straightforward way.
  - In any pair of communicating applications,
    - one side must start execution and wait (indefinitely) –server
    - the other side to contact it –client

Terminologies in Client-Server Model

- **Client** – an application that initiates peer-to peer communication.
  - Each time client executes, it contacts a server, send a request, and awaits a response.
- **Server** – any program that waits for incoming communication requests from a client.
  - The server receives a client's request, performs the necessary computation, and returns the result to the client.

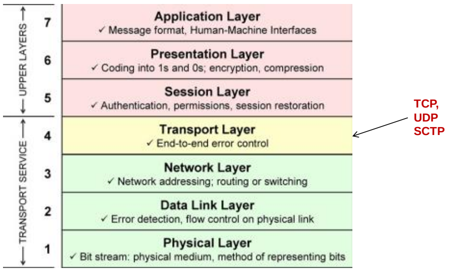
Terminologies in Client-Server Model

- A server contains code that handles many issues that can be grouped under security:
  - **Authentication** – verifying the identity of the client
  - **Authorization** – check the permission for the client to access the service
  - **Data security** – guarantee that data is not unintentionally revealed
  - **Privacy** – keep information from unauthorized access
  - **Protection** – guarantee that network application cannot abuse system resource

Connectionless vs. Connection-Oriented

- When programmer designs client-server application software, they must choose types of interaction.
  - **Connectionless** – use **U**ser **D**atagram **P**rotocol (**UDP**)
  - **Connection-Oriented** – use **T**ransmission **C**ontrol **P**rotocol (**TCP**) or use **S**tream **C**ontrol **T**ransmission Protocol (**SCTP**)

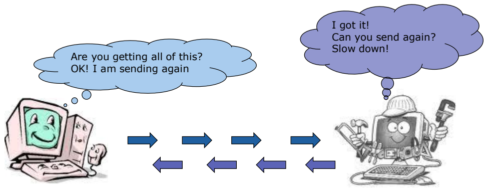
Connectionless vs. Connection-Oriented


UPPER LAYERS
TRANSPORT SERVICE
7
Application Layer
✓ Message format, Human-Machine Interfaces
6
Presentation Layer
✓ Coding into 1s and 0s; encryption, compression
5
Session Layer
✓ Authentication, permissions, session restoration
4
Transport Layer
✓ End-to-end error control
3
Network Layer
✓ Network addressing; routing or switching
2
Data Link Layer
✓ Error detection, flow control on physical link
1
Physical Layer
✓ Bit stream: physical medium, method of representing bits
TCP,
UDP
SCTP

Connectionless vs. Connection-Oriented

- TCP provide full reliability by taking care of all transmission problems:
  - **Verifies data arrives**- if not, retransmit segment.
  - **Error checking** – by checking checksum, guarantee error free transmission.
  - TCP uses sequence number to ensure the data arrives in order and eliminate duplicate copy.
  - TCP provides **flow control** to ensure that the sender does not transmit data faster than receiver can consume it.

Connectionless vs. Connection-Oriented

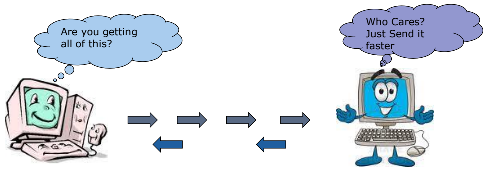


TCP: Connection- Oriented

Connectionless vs. Connection-Oriented

- UDP do not have any guarantees about reliable delivery
  - When a client sends a request, the request may be lost, duplicated, delayed or delivered out of order.
  - A response from the server to client also might be lost, duplicated, delayed or delivered out of order.
  - If a programmer use UDP for client-server application, application software itself must contain code to take care of such errors.

Connectionless vs. Connection-Oriented



UDP: Connectionless

Connectionless vs. Connection-Oriented

- Program only use UDP
  - If the application Protocol handles reliability,
  - Application requires hardware broadcast or multicast, or the overhead for reliability is not needed.
- UDP is commonly used for streaming audio and video. Streaming media such as Windows Media audio files (.WMA) , Real Player (.RM), and others use UDP because it offers speed!
- The reason UDP is faster than TCP is because there is no form of flow control or error correction.

Stateless Vs. Stateful Server

- The stateless server does not keep a track of the user's identity or actions performed on the page. So every time, the client has to prove the identity to gain access.
- A **stateless server** is treats each request as an independent transaction that is unrelated to any previous request.
  - The motivation of a stateless server lies in protocol reliability.
    - State information in a server can become incorrect if messages are lost, duplicated, or delivered out of order.
    - It cause server might respond incorrectly.

Stateless Vs. Stateful Server

Ex) Stateless file server

- Since server does not save any information regarding a client request, each request from a client must specify complete information: operation (read/write), name of file, position in the file, number of bytes to transfer , and data (for writing).

Stateless Vs. Stateful Server

- **A stateful server** is a server which keeps data about previous requests for a service which can then be used for subsequent requests.
  - Keeping a small amount of information can reduce the size of messages that the client and server exchange, and can allow the server to respond to requests quickly.
  - If the client state change by error (ex. Client site support multiple IP, Client site change IP address), stateful server might send wrong response.

Stateless Vs. Stateful Server

Ex Sateful file server)

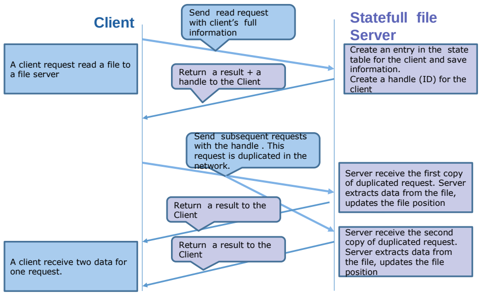
- A stateful file server maintains a table that holds information about each client and the file currently being accessed.
- Once a client begins accessing a file (read or write), the server retain information: operation, name of file, and position in the file.
- After initial request to read from a file, subsequent read request messages only need to contain one field: the number of bytes to read.
- Once client begins writing data, each subsequent write request messages only need to contain two fields: the number of bytes to write and data.

Stateless Vs. Stateful Server

Ex Sateful file server Cont.)

- Consider what will happen if the network duplicates a read request from a client to sateful file server.
  - Server updates its notion of file position each time the client extract data from a file.
  - Server will receive two copies of request since the read request is duplicated.
  - When the first copy arrives, server extracts data from the file, updates the file position and returns the result to client. When the second copy arrives, server extracts data again from the file, updates the file position again and returns the result to the client.
  - Client will receive two different results for one request. Client might discard it or it may reports an error since it received two different results for one request.

Stateless Vs. Stateful Server



Client

Statefull file Server

A client request read a file to a file server

Send read request with client's full information

Create an entry in the state table for the client and save information.
Create a handle (ID) for the client

Return a result + a handle to the Client

Send subsequent requests with the handle . This request is duplicated in the network.

Server receive the first copy of duplicated request. Server extracts data from the file, updates the file position

Return a result to the Client

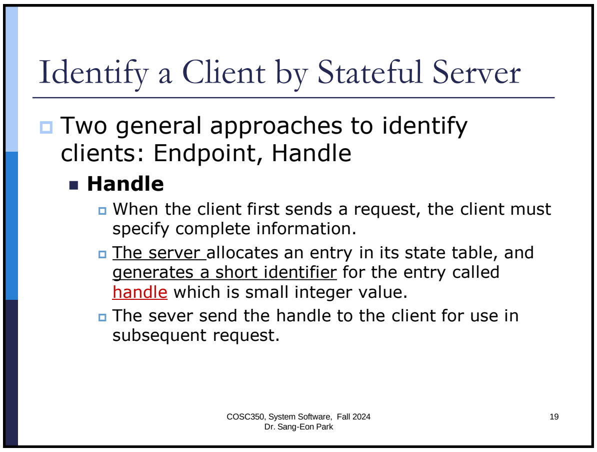
Server receive the second copy of duplicated request. Server extracts data from the file, updates the file position

Return a result to the Client

A client receive two data for one request.

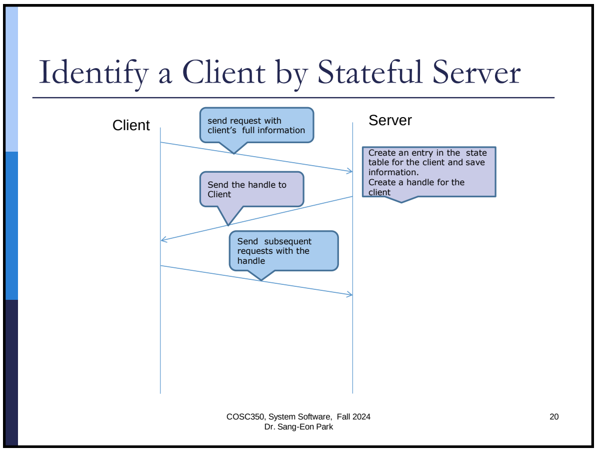
Identify a Client by Stateful Server

- Two general approaches to identify clients: **Endpoint**, **Handle**
  - **Endpoints**
    - The sever ask the underlying transport protocol to provide identifying information when a request arrives (IP address and Port number used by client).
    - Then, the server uses the endpoint information to identify a particular slot in its state table.
    - Unfortunately, endpoint information can change (ex. A client need to open a new TCP connection due to a network failure) .



Identify a Client by Stateful Server

- Two general approaches to identify clients: Endpoint, Handle
  - **Handle**
    - When the client first sends a request, the client must specify complete information.
    - The server allocates an entry in its state table, and generates a short identifier for the entry called handle which is small integer value.
    - The sever send the handle to the client for use in subsequent request.



Identify a Client by Stateful Server

Client

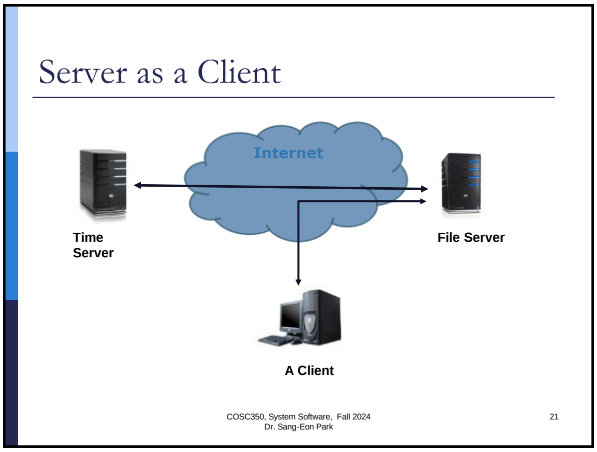
Server

send request with client's full information

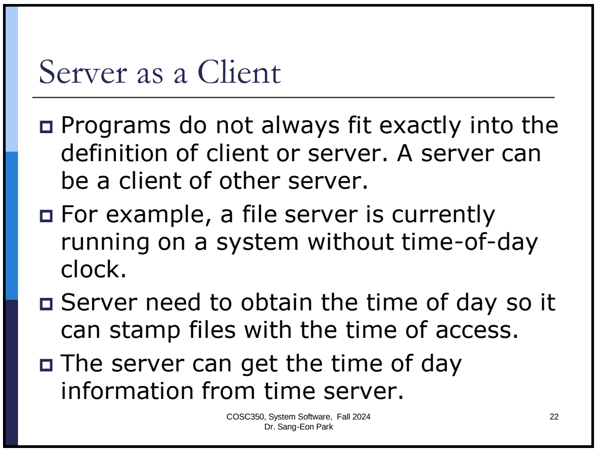
Create an entry in the state table for the client and save information. Create a handle for the client

Send the handle to Client

Send subsequent requests with the handle



Server as a Client



Server as a Client

- Programs do not always fit exactly into the definition of client or server. A server can be a client of other server.
- For example, a file server is currently running on a system without time-of-day clock.
- Server need to obtain the time of day so it can stamp files with the time of access.
- The server can get the time of day information from time server.